REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-00-

0791

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT
	December, 1997	

4. TITLE AND SUBTITLE
1997 Summer Research Program (SRP), High School Apprenticeship Program (HSAP), Final Reports, Volume 14, Rome Laboratory

5. FUNDING NUMBERS
F49620-93-C-0063

6. AUTHOR(S)
Gary Moore

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Research & Development Laboratories (RDL)
5800 Uplander Way
Culver City, CA 90230-6608

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research (AFOSR)
801 N. Randolph St.
Arlington, VA 22203-1977

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The United States Air Force Summer Research Program (USAF-SRP) is designed to introduce university, college, and technical institute faculty members, graduate students, and high school students to Air Force research. This is accomplished by the faculty members (Summer Faculty Research Program, (SFRP)), graduate students (Graduate Student Research Program (GSRP)), and high school students (High School Apprenticeship Program (HSAP)) being selected on a nationally advertised competitive basis during the summer intersession period to perform research at Air Force Research Laboratory (AFRL) Technical Directorates, Air Force Air Logistics Centers (ALC), and other AF Laboratories. This volume consists of a program overview, program management statistics, and the final technical reports from the HSAP participants at the Rome Laboratory.

14. SUBJECT TERMS
Air Force Research, Air Force, Engineering, Laboratories, Reports, Summer, Universities, Faculty, Graduate Student, High School Student

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

UNITED STATES AIR FORCE

SUMMER RESEARCH PROGRAM -- 1997

HIGH SCHOOL APPRENTICESHIP PROGRAM FINAL REPORTS

VOLUME 14

ROME LABORATORY

RESEARCH & DEVELOPMENT LABORATORIES

5800 Uplander Way

Culver City, CA 90230-6608

Program Director, RDL
Gary Moore

Program Manager, AFOSR
Major Linda Steel-Goodwin

Program Manager, RDL
Scott Licoscos

Program Administrator, RDL
Johnetta Thompson

Program Administrator, RDL
Rebecca Kelly-Clemmons

Submitted to:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

Bolling Air Force Base

Washington, D.C.

December 1997

PREFACE

Reports in this volume are numbered consecutively beginning with number 1. Each report is paginated with the report number followed by consecutive page numbers, e.g., 1-1, 1-2, 1-3; 2-1, 2-2, 2-3.

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

SRP Final Report Table of Contents

1. INTRODUCTION

The Summer Research Program (SRP), sponsored by the Air Force Office of Scientific Research (AFOSR), offers paid opportunities for university faculty, graduate students, and high school students to conduct research in U.S. Air Force research laboratories nationwide during the summer.

Introduced by AFOSR in 1978, this innovative program is based on the concept of teaming academic researchers with Air Force scientists in the same disciplines using laboratory facilities and equipment not often available at associates' institutions.

The Summer Faculty Research Program (SFRP) is open annually to approximately 150 faculty members with at least two years of teaching and/or research experience in accredited U.S. colleges, universities, or technical institutions. SFRP associates must be either U.S. citizens or permanent residents.

The Graduate Student Research Program (GSRP) is open annually to approximately 100 graduate students holding a bachelor's or a master's degree; GSRP associates must be U.S. citizens enrolled full time at an accredited institution.

The High School Apprentice Program (HSAP) annually selects about 125 high school students located within a twenty mile commuting distance of participating Air Force laboratories.

AFOSR also offers its research associates an opportunity, under the Summer Research Extension Program (SREP), to continue their AFOSR-sponsored research at their home institutions through the award of research grants. In 1994 the maximum amount of each grant was increased from $20,000 to $25,000, and the number of AFOSR-sponsored grants decreased from 75 to 60. A separate annual report is compiled on the SREP.

The numbers of projected summer research participants in each of the three categories and SREP "grants" are usually increased through direct sponsorship by participating laboratories.

AFOSR's SRP has well served its objectives of building critical links between Air Force research laboratories and the academic community, opening avenues of communications and forging new research relationships between Air Force and academic technical experts in areas of national interest, and strengthening the nation's efforts to sustain careers in science and engineering. The success of the SRP can be gauged from its growth from inception (see Table 1) and from the favorable responses the 1997 participants expressed in end-of-tour SRP evaluations (Appendix B).

AFOSR contracts for administration of the SRP by civilian contractors. The contract was first awarded to Research & Development Laboratories (RDL) in September 1990. After completion of the

1990 contract, RDL (in 1993) won the recompetition for the basic year and four 1-year options.

2. PARTICIPATION IN THE SUMMER RESEARCH PROGRAM

The SRP began with faculty associates in 1979; graduate students were added in 1982 and high school students in 1986. The following table shows the number of associates in the program each year.

YEAR	SRP Participation, by Year			TOTAL
	SFRP	GSRP	HSAP	
1979	70			70
1980	87			87
1981	87			87
1982	91	17		108
1983	101	53		154
1984	152	84		236
1985	154	92		246
1986	158	100	42	300
1987	159	101	73	333
1988	153	107	101	361
1989	168	102	103	373
1990	165	121	132	418
1991	170	142	132	444
1992	185	121	159	464
1993	187	117	136	440
1994	192	117	133	442
1995	190	115	137	442
1996	188	109	138	435
1997	148	98	140	427

Beginning in 1993, due to budget cuts, some of the laboratories weren't able to afford to fund as many associates as in previous years. Since then, the number of funded positions has remained fairly constant at a slightly lower level.

3. RECRUITING AND SELECTION

The SRP is conducted on a nationally advertised and competitive-selection basis. The advertising for faculty and graduate students consisted primarily of the mailing of 8,000 52-page SRP brochures to chairpersons of departments relevant to AFOSR research and to administrators of grants in accredited universities, colleges, and technical institutions. Historically Black Colleges and Universities (HBCUs) and Minority Institutions (MIs) were included. Brochures also went to all participating USAF laboratories, the previous year's participants, and numerous individual requesters (over 1000 annually).

RDL placed advertisements in the following publications: *Black Issues in Higher Education, Winds of Change,* and *IEEE Spectrum.* Because no participants list either *Physics Today* or *Chemical & Engineering News* as being their source of learning about the program for the past several years, advertisements in these magazines were dropped, and the funds were used to cover increases in brochure printing costs.

High school applicants can participate only in laboratories located no more than 20 miles from their residence. Tailored brochures on the HSAP were sent to the head counselors of 180 high schools in the vicinity of participating laboratories, with instructions for publicizing the program in their schools. High school students selected to serve at Wright Laboratory's Armament Directorate (Eglin Air Force Base, Florida) serve eleven weeks as opposed to the eight weeks normally worked by high school students at all other participating laboratories.

Each SFRP or GSRP applicant is given a first, second, and third choice of laboratory. High school students who have more than one laboratory or directorate near their homes are also given first, second, and third choices.

Laboratories make their selections and prioritize their nominees. AFOSR then determines the number to be funded at each laboratory and approves laboratories' selections.

Subsequently, laboratories use their own funds to sponsor additional candidates. Some selectees do not accept the appointment, so alternate candidates are chosen. This multi-step selection procedure results in some candidates being notified of their acceptance after scheduled deadlines. The total applicants and participants for 1997 are shown in this table.

1997 Applicants and Participants			
PARTICIPANT CATEGORY	TOTAL APPLICANTS	SELECTEES	DECLINING SELECTEES
SFRP	490	188	32
(HBCU/MI)	(0)	(0)	(0)
GSRP	202	98	9
(HBCU/MI)	(0)	(0)	(0)
HSAP	433	140	14
TOTAL	1125	426	55

4. SITE VISITS

During June and July of 1997, representatives of both AFOSR/NI and RDL visited each participating laboratory to provide briefings, answer questions, and resolve problems for both laboratory personnel and participants. The objective was to ensure that the SRP would be as constructive as possible for all participants. Both SRP participants and RDL representatives found these visits beneficial. At many of the laboratories, this was the only opportunity for all participants to meet at one time to share their experiences and exchange ideas.

5. HISTORICALLY BLACK COLLEGES AND UNIVERSITIES AND MINORITY INSTITUTIONS (HBCU/MIs)

Before 1993, an RDL program representative visited from seven to ten different HBCU/MIs annually to promote interest in the SRP among the faculty and graduate students. These efforts were marginally effective, yielding a doubling of HBCI/MI applicants. In an effort to achieve AFOSR's goal of 10% of all applicants and selectees being HBCU/MI qualified, the RDL team decided to try other avenues of approach to increase the number of qualified applicants. Through the combined efforts of the AFOSR Program Office at Bolling AFB and RDL, two very active minority groups were found, HACU (Hispanic American Colleges and Universities) and AISES (American Indian Science and Engineering Society). RDL is in communication with representatives of each of these organizations on a monthly basis to keep up with the their activities and special events. Both organizations have widely-distributed magazines/quarterlies in which RDL placed ads.

Since 1994 the number of both SFRP and GSRP HBCU/MI applicants and participants has increased ten-fold, from about two dozen SFRP applicants and a half dozen selectees to over 100 applicants and two dozen selectees, and a half-dozen GSRP applicants and two or three selectees to 18 applicants and 7 or 8 selectees. Since 1993, the SFRP had a two-fold applicant increase and a two-fold selectee increase. Since 1993, the GSRP had a three-fold applicant increase and a three to four-fold increase in selectees.

In addition to RDL's special recruiting efforts, AFOSR attempts each year to obtain additional funding or use leftover funding from cancellations the past year to fund HBCU/MI associates. This year, 5 HBCU/MI SFRPs declined after they were selected (and there was no one qualified to replace them with). The following table records HBCU/MI participation in this program.

SRP HBCU/MI Participation, By Year				
YEAR	SFRP		GSRP	
	Applicants	Participants	Applicants	Participants
1985	76	23	15	11
1986	70	18	20	10
1987	82	32	32	10
1988	53	17	23	14
1989	39	15	13	4
1990	43	14	17	3
1991	42	13	8	5
1992	70	13	9	5
1993	60	13	6	2
1994	90	16	11	6
1995	90	21	20	8
1996	119	27	18	7

6. SRP FUNDING SOURCES

Funding sources for the 1997 SRP were the AFOSR-provided slots for the basic contract and laboratory funds. Funding sources by category for the 1997 SRP selected participants are shown here.

1997 SRP FUNDING CATEGORY	SFRP	GSRP	HSAP
AFOSR Basic Allocation Funds	141	89	123
USAF Laboratory Funds	48	9	17
HBCU/MI By AFOSR (Using Procured Addn'l Funds)	0	0	N/A
TOTAL	9	98	140

SFRP - 188 were selected, but thirty two canceled too late to be replaced.
GSRP - 98 were selected, but nine canceled too late to be replaced.
HSAP - 140 were selected, but fourteen canceled too late to be replaced.

7. COMPENSATION FOR PARTICIPANTS

Compensation for SRP participants, per five-day work week, is shown in this table.

1997 SRP Associate Compensation

PARTICIPANT CATEGORY	1991	1992	1993	1994	1995	1996	1997
Faculty Members	$690	$718	$740	$740	$740	$770	$770
Graduate Student (Master's Degree)	$425	$442	$455	$455	$455	$470	$470
Graduate Student (Bachelor's Degree)	$365	$380	$391	$391	$391	$400	$400
High School Student (First Year)	$200	$200	$200	$200	$200	$200	$200
High School Student (Subsequent Years)	$240	$240	$240	$240	$240	$240	$240

The program also offered associates whose homes were more than 50 miles from the laboratory an expense allowance (seven days per week) of $50/day for faculty and $40/day for graduate students. Transportation to the laboratory at the beginning of their tour and back to their home destinations at the end was also reimbursed for these participants. Of the combined SFRP and GSRP associates, 65 % (194 out of 286) claimed travel reimbursements at an average round-trip cost of $776.

Faculty members were encouraged to visit their laboratories before their summer tour began. All costs of these orientation visits were reimbursed. Forty-three percent (85 out of 188) of faculty associates took orientation trips at an average cost of $388. By contrast, in 1993, 58 % of SFRP associates took

orientation visits at an average cost of $685; that was the highest percentage of associates opting to take an orientation trip since RDL has administered the SRP, and the highest average cost of an orientation trip. These 1993 numbers are included to show the fluctuation which can occur in these numbers for planning purposes.

Program participants submitted biweekly vouchers countersigned by their laboratory research focal point, and RDL issued paychecks so as to arrive in associates' hands two weeks later.

This is the second year of using direct deposit for the SFRP and GSRP associates. The process went much more smoothly with respect to obtaining required information from the associates, only 7% of the associates' information needed clarification in order for direct deposit to properly function as opposed to 10% from last year. The remaining associates received their stipend and expense payments via checks sent in the US mail.

HSAP program participants were considered actual RDL employees, and their respective state and federal income tax and Social Security were withheld from their paychecks. By the nature of their independent research, SFRP and GSRP program participants were considered to be consultants or independent contractors. As such, SFRP and GSRP associates were responsible for their own income taxes, Social Security, and insurance.

8. CONTENTS OF THE 1997 REPORT

The complete set of reports for the 1997 SRP includes this program management report (Volume 1) augmented by fifteen volumes of final research reports by the 1997 associates, as indicated below:

1997 SRP Final Report Volume Assignments

LABORATORY	SFRP	GSRP	HSAP
Armstrong	2	7	12
Phillips	3	8	13
Rome	4	9	14
Wright	5A, 5B	10	15
AEDC, ALCs, WHMC	6	11	16

APPENDIX A – PROGRAM STATISTICAL SUMMARY

A. Colleges/Universities Represented

Selected SFRP associates represented 169 different colleges, universities, and institutions, GSRP associates represented 95 different colleges, universities, and institutions.

B. States Represented

SFRP -Applicants came from 47 states plus Washington D.C. Selectees represent 44 states.

GSRP - Applicants came from 44 states. Selectees represent 32 states.

HSAP - Applicants came from thirteen states. Selectees represent nine states.

Total Number of Participants	
SFRP	189
GSRP	97
HSAP	140
TOTAL	426

Degrees Represented			
	SFRP	GSRP	TOTAL
Doctoral	184	0	184
Master's	2	41	43
Bachelor's	0	56	56
TOTAL	186	97	298

SFRP Academic Titles	
Assistant Professor	64
Associate Professor	70
Professor	40
Instructor	0
Chairman	1
Visiting Professor	1
Visiting Assoc. Prof.	1
Research Associate	9
TOTAL	186

Source of Learning About the SRP		
Category	Applicants	Selectees
Applied/participated in prior years	28%	34%
Colleague familiar with SRP	19%	16%
Brochure mailed to institution	23%	17%
Contact with Air Force laboratory	17%	23%
IEEE Spectrum	2%	1%
BIIHE	1%	1%
Other source	10%	8%
TOTAL	100%	100%

APPENDIX B – SRP EVALUATION RESPONSES

1. OVERVIEW

Evaluations were completed and returned to RDL by four groups at the completion of the SRP. The number of respondents in each group is shown below.

Table B-1. Total SRP Evaluations Received

Evaluation Group	Responses
SFRP & GSRPs	275
HSAPs	113
USAF Laboratory Focal Points	84
USAF Laboratory HSAP Mentors	6

All groups indicate unanimous enthusiasm for the SRP experience.

The summarized recommendations for program improvement from both associates and laboratory personnel are listed below:

A. Better preparation on the labs' part prior to associates' arrival (i.e., office space, computer assets, clearly defined scope of work).

B. Faculty Associates suggest higher stipends for SFRP associates.

C. Both HSAP Air Force laboratory mentors and associates would like the summer tour extended from the current 8 weeks to either 10 or 11 weeks; the groups state it takes +- 6 weeks just to get high school students up-to-speed on what's going on at laboratory. (Note: this same argument was used to raise the faculty and graduate student participation time a few years ago.)

2. 1997 USAF LABORATORY FOCAL POINT (LFP) EVALUATION RESPONSES

The summarized results listed below are from the 84 LFP evaluations received.

1. LFP evaluations received and associate preferences:

Table B-2. Air Force LFP Evaluation Responses (By Type)

		How Many Associates Would You Prefer To Get ? (% Response)											
		SFRP				GSRP (w/Univ Professor)				GSRP (w/o Univ Professor)			
Lab	Evals Recv'd	0	1	2	3+	0	1	2	3+	0	1	2	3+
AEDC	0	-	-	-	-	-	-	-	-	-	-	-	-
WHMC	0	-	-	-	-	-	-	-	-	-	-	-	-
AL	7	28	28	28	14	54	14	28	0	86	0	14	0
USAFA	1	0	100	0	0	100	0	0	0	0	100	0	0
PL	25	40	40	16	4	88	12	0	0	84	12	4	0
RL	5	60	40	0	0	80	10	0	0	100	0	0	0
WL	46	30	43	20	6	78	17	4	0	93	4	2	0
Total	84	32%	50%	13%	5%	80%	11%	6%	0%	73%	23%	4%	0%

LFP Evaluation Summary. The summarized responses, by laboratory, are listed on the following page. LFPs were asked to rate the following questions on a scale from 1 (below average) to 5 (above average).

2. LFPs involved in SRP associate application evaluation process:
   a. Time available for evaluation of applications:
   b. Adequacy of applications for selection process:
3. Value of orientation trips:
4. Length of research tour:
5. a. Benefits of associate's work to laboratory:
   b. Benefits of associate's work to Air Force:
6. a. Enhancement of research qualifications for LFP and staff:
   b. Enhancement of research qualifications for SFRP associate:
   c. Enhancement of research qualifications for GSRP associate:
7. a. Enhancement of knowledge for LFP and staff:
   b. Enhancement of knowledge for SFRP associate:
   c. Enhancement of knowledge for GSRP associate:
8. Value of Air Force and university links:
9. Potential for future collaboration:
10. a. Your working relationship with SFRP:
    b. Your working relationship with GSRP:
11. Expenditure of your time worthwhile:

(Continued on next page)

12. Quality of program literature for associate:
13. a. Quality of RDL's communications with you:
    b. Quality of RDL's communications with associates:
14. Overall assessment of SRP:

Table B-3. Laboratory Focal Point Reponses to above questions

	AEDC	*AL*	*USAFA*	*PL*	*RL*	*WHMC*	*WL*
# Evals Recv'd	0	7	1	14	5	0	46
Question #							
2	-	86 %	0 %	88 %	80 %	-	85 %
2a	-	4.3	n/a	3.8	4.0	-	3.6
2b	-	4.0	n/a	3.9	4.5	-	4.1
3	-	4.5	n/a	4.3	4.3	-	3.7
4	-	4.1	4.0	4.1	4.2	-	3.9
5a	-	4.3	5.0	4.3	4.6	-	4.4
5b	-	4.5	n/a	4.2	4.6	-	4.3
6a	-	4.5	5.0	4.0	4.4	-	4.3
6b	-	4.3	n/a	4.1	5.0	-	4.4
6c	-	3.7	5.0	3.5	5.0	-	4.3
7a	-	4.7	5.0	4.0	4.4	-	4.3
7b	-	4.3	n/a	4.2	5.0	-	4.4
7c	-	4.0	5.0	3.9	5.0	-	4.3
8	-	4.6	4.0	4.5	4.6	-	4.3
9	-	4.9	5.0	4.4	4.8	-	4.2
10a	-	5.0	n/a	4.6	4.6	-	4.6
10b	-	4.7	5.0	3.9	5.0	-	4.4
11	-	4.6	5.0	4.4	4.8	-	4.4
12	-	4.0	4.0	4.0	4.2	-	3.8
13a	-	3.2	4.0	3.5	3.8	-	3.4
13b	-	3.4	4.0	3.6	4.5	-	3.6
14	-	4.4	5.0	4.4	4.8	-	4.4

3. 1997 SFRP & GSRP EVALUATION RESPONSES

The summarized results listed below are from the 257 SFRP/GSRP evaluations received.

Associates were asked to rate the following questions on a scale from 1 (below average) to 5 (above average) - by Air Force base results and over-all results of the 1997 evaluations are listed after the questions.

1. The match between the laboratories research and your field:
2. Your working relationship with your LFP:
3. Enhancement of your academic qualifications:
4. Enhancement of your research qualifications:
5. Lab readiness for you: LFP, task, plan:
6. Lab readiness for you: equipment, supplies, facilities:
7. Lab resources:
8. Lab research and administrative support:
9. Adequacy of brochure and associate handbook:
10. RDL communications with you:
11. Overall payment procedures:
12. Overall assessment of the SRP:
13. a. Would you apply again?
    b. Will you continue this or related research?
14. Was length of your tour satisfactory?
15. Percentage of associates who experienced difficulties in finding housing:
16. Where did you stay during your SRP tour?
    a. At Home:
    b. With Friend:
    c. On Local Economy:
    d. Base Quarters:
17. Value of orientation visit:
    a. Essential:
    b. Convenient:
    c. Not Worth Cost:
    d. Not Used:

SFRP and GSRP associate's responses are listed in tabular format on the following page.

Table B-4. 1997 SFRP & GSRP Associate Responses to SRP Evaluation

	Arnold	Brooks	Edwards	Eglin	Griffis	Hanscom	Kelly	Kirtland	Lackland	Robins	Tyndall	WPAFB	average
# res	6	48	6	14	31	19	3	32	1	2	10	85	257
1	4.8	4.4	4.6	4.7	4.4	4.9	4.6	4.6	5.0	5.0	4.0	4.7	4.6
2	5.0	4.6	4.1	4.9	4.7	4.7	5.0	4.7	5.0	5.0	4.6	4.8	4.7
3	4.5	4.4	4.0	4.6	4.3	4.2	4.3	4.4	5.0	5.0	4.5	4.3	4.4
4	4.3	4.5	3.8	4.6	4.4	4.4	4.3	4.6	5.0	4.0	4.4	4.5	4.5
5	4.5	4.3	3.3	4.8	4.4	4.5	4.3	4.2	5.0	5.0	3.9	4.4	4.4
6	4.3	4.3	3.7	4.7	4.4	4.5	4.0	3.8	5.0	5.0	3.8	4.2	4.2
7	4.5	4.4	4.2	4.8	4.5	4.3	4.3	4.1	5.0	5.0	4.3	4.3	4.4
8	4.5	4.6	3.0	4.9	4.4	4.3	4.3	4.5	5.0	5.0	4.7	4.5	4.5
9	4.7	4.5	4.7	4.5	4.3	4.5	4.7	4.3	5.0	5.0	4.1	4.5	4.5
10	4.2	4.4	4.7	4.4	4.1	4.1	4.0	4.2	5.0	4.5	3.6	4.4	4.3
11	3.8	4.1	4.5	4.0	3.9	4.1	4.0	4.0	3.0	4.0	3.7	4.0	4.0
12	5.7	4.7	4.3	4.9	4.5	4.9	4.7	4.6	5.0	4.5	4.6	4.5	4.6
Numbers below are percentages													
13a	83	90	83	93	87	75	100	81	100	100	100	86	87
13b	100	89	83	100	94	98	100	94	100	100	100	94	93
14	83	96	100	90	87	80	100	92	100	100	70	84	88
15	17	6	0	33	20	76	33	25	0	100	20	8	39
16a	-	26	17	9	38	23	33	4	-	-	-	30	
16b	100	33	-	40	-	8	-	-	-	-	36	2	
16c	-	41	83	40	62	69	67	96	100	100	64	68	
16d	-	-	-	-	-	-	-	-	-	-	-	0	
17a	-	33	100	17	50	14	67	39	-	50	40	31	35
17b	-	21	-	17	10	14	-	24	-	50	20	16	16
17c	-	-	-	-	10	7	-	-	-	-	-	2	3
17d	100	46	-	66	30	69	33	37	100	-	40	51	46

4. 1997 USAF LABORATORY HSAP MENTOR EVALUATION RESPONSES

Not enough evaluations received (5 total) from Mentors to do useful summary.

5. 1997 HSAP EVALUATION RESPONSES

The summarized results listed below are from the 113 HSAP evaluations received.

HSAP apprentices were asked to rate the following questions on a scale from 1 (below average) to 5 (above average)

1. Your influence on selection of topic/type of work.
2. Working relationship with mentor, other lab scientists.
3. Enhancement of your academic qualifications.
4. Technically challenging work.
5. Lab readiness for you: mentor, task, work plan, equipment.
6. Influence on your career.
7. Increased interest in math/science.
8. Lab research & administrative support.
9. Adequacy of RDL's Apprentice Handbook and administrative materials.
10. Responsiveness of RDL communications.
11. Overall payment procedures.
12. Overall assessment of SRP value to you.
13. Would you apply again next year? Yes (92 %)
14. Will you pursue future studies related to this research? Yes (68 %)
15. Was Tour length satisfactory? Yes (82 %)

	Arnold	Brooks	Edwards	Eglin	Griffiss	Hanscom	Kirtland	Tyndall	WPAFB	Totals
# resp	5	19	7	15	13	2	7	5	40	113
1	2.8	3.3	3.4	3.5	3.4	4.0	3.2	3.6	3.6	3.4
2	4.4	4.6	4.5	4.8	4.6	4.0	4.4	4.0	4.6	4.6
3	4.0	4.2	4.1	4.3	4.5	5.0	4.3	4.6	4.4	4.4
4	3.6	3.9	4.0	4.5	4.2	5.0	4.6	3.8	4.3	4.2
5	4.4	4.1	3.7	4.5	4.1	3.0	3.9	3.6	3.9	4.0
6	3.2	3.6	3.6	4.1	3.8	5.0	3.3	3.8	3.6	3.7
7	2.8	4.1	4.0	3.9	3.9	5.0	3.6	4.0	4.0	3.9
8	3.8	4.1	4.0	4.3	4.0	4.0	4.3	3.8	4.3	4.2
9	4.4	3.6	4.1	4.1	3.5	4.0	3.9	4.0	3.7	3.8
10	4.0	3.8	4.1	3.7	4.1	4.0	3.9	2.4	3.8	3.8
11	4.2	4.2	3.7	3.9	3.8	3.0	3.7	2.6	3.7	3.8
12	4.0	4.5	4.9	4.6	4.6	5.0	4.6	4.2	4.3	4.5
Numbers below are percentages										
13	60%	95%	100%	100%	85%	100%	100%	100%	90%	92%
14	20%	80%	71%	80%	54%	100%	71%	80%	65%	68%
15	100%	70%	71%	100%	100%	50%	86%	60%	80%	82%

HTML Computer Language

Kristine Angell

Camden High School
Oswego St.
Camden, NY 13316

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

HTML Computer Language

Kristine Angell
Camden High School

Abstract

HTML is a computer language used to create Web pages for the Internet. The HTML file is created on a word processing program, and is viewed graphically through the Internet. The HTML file consists of tag commands that create a document know as a Web page. The basic structure of an HTML file consists of the head and body. Within this structure exists a plethora of possible tags to better organize data and create style. With HTML there is also the ability to link sites together with similar information sources through the Internet.

HTML Computer Language

Kristine Angell

1. INTRODUCTION

HTML is a computer language used in the creation of Internet Web pages. Through the interconnection of the Internet the information on Web pages can be accessed by anyone in the world. HyperText Markup Language (**HTML**) is a hypertext variant of the Standard Generalized Markup Language (SBML) which marks a document's parts so that the browser can tell what they are. HTML can be read by any graphical browser, it does this by declaring the structure of the document and leaving the specifics of formatting to the browsing software. HTML files are always plain text with no special characters or binary code. These types of files are easy to exchange via Internet connections.

2.HTML

HTML is not a programming language, which allows the computer to perform an action. HTML let's you declare parts of a document. This document can then be viewed as it is declared, though interpretation is left up to the individual client programs. A word processing program specifies precisely how a given unit of text should appear. HTML varies from a word processing program with the variable of interpretation. It declares the parts of the text and allows the graphical browser to determine the actual on-screen document.

HTML is displayed in a text editor or word processing program for creation, editing and updating. An HTML document is an ordinary text file (ASCII file) that contains texts and tags. The file name itself must end with the extension .html (with Windows .htm is used).

A tag is an instruction that an HTML capable browser can recognize and process; it doesn't appear in the document when viewing the file on the Internet.

3. SECTIONS OF AN HTML DOCUMENT

A. TAGS

Tags tell the browser program what kind of text is found between the paired tags. All HTML tags are encased in '<'and '>' signs. Inside these signs are the commands for interpreting the given text. Most HTML tags have a beginning and end tag. The beginning tag is the tag itself, the closing tag is the beginning tag with a forward slash.

Ex. <title> *text* </title>

Some tags are independent and do not require a corresponding end tag. These tags are know as empty tags.

Ex. <hr>

HTML is comprised of 2 separate sections: the head and the body. The latter is where most, if not

all, writing will take place. The actions for a Web page result from tags used in the text. Two tags are

required: <html> to open the page and </html> to close it. The <head>,</head>; <body>, </body>; <title>, </title>; and <address>, </address> tags are also essential parts of an HTML document, and all files should minimally contain these 5 tags.

3. B. HEAD

The head is the first section of an HTML document. A properly designed HTML document will contain a head and within the head a title. The base and link commands are optional. *Figure A* lists the possible tags and functions.

FIGURE A:

TAG	COMMAND
<head> </head>	declares the head
<title> </title>	labels title
<base>	provides absolute URL base for any relative URL links in the document
<link>	links document to creator and/or another document

3. C. BODY

In the body lies the text that's shown inside the browser window. There are many commands to use to change the way the text looks. Within the tags the case of the commands is irrelevant. Thus '<body>', '<BODY>' and '<BoDy>' are all equivalent. *Figure B* is a list of basic commands and their functions that are used in the body of an HTML file.

FIGURE B:

TAG	COMMAND
<address> </address>	address of creator
<applet> </applet>	identifies and invokes a JAVA application
	bolds the font
<banner> </banner>	identifies corporate logos which should not be scrolled with the rest of the document
<blink> </blink>	blinking font
<blockquote> </blockquote>	indents sections of text --like a quotation
<body> </body>	declares body of text
 	adds on space or return between lines
<caption> </caption>	labels a figure or table
	centers the font
<cite> </cite>	changes font to that of a citation
<fn> </fn>	changes font to that of a footnote
<font color = blue>	defines font color

TAG	COMMAND
<font size = #>	defines font size
<h#> </h#>	declares headings (1-6) 1 being the largest
<hr>	creates a horizontal line
	italicizes the font
<img src = "file">	declares and image
<p>	adds a break between paragraphs
<param>	defines general parameters to be passed to APPLET applications
<person> </person>	declares people, to be extracted automatically when indexing programs
<pre> </pre>	seperates text to be displayed exactly as it is typed into the html file
<s> </s>	places a strikeout line through the font
	subscripts the font
	superscripts the font
<u> </u>	underlines the font

An HTML file through the word processing window would appear as ex.1

Ex 1.

```
<html>
<head><title>demo page</title></head>
<body> <font size = 5 color = black>
<b><center> KRISTINE ANGELL</center></b>
Born in the year <u> 1979</u> <br>graduated in the year <I> 1997</I>
<p>Today is August 11, 1997<blockquote> "You belong among the wildflowers,<br>
you belong somewhere close to me, <br> you belong among the wildflowers,<br> you belong
somewhere you feel free
<cite>--Tom Petty</cite></blockquote>
<address>Kris@non-existant </address>
</body>
</html>
```

This HTML file viewed through Netscape would appear as ex 2.

Ex 2.

KRISTINE ANGELL

Born in the year 1979

graduated in the year *1997*

Today is August 11,1997

"You belong among the wildflowers, you belong somewhere close to me,
you belong among the wildflowers, you belong somewhere you feel free"
--Tom Petty

Kris@non-existant

3. D . ADVANCED TAGS

3. D. 1 attributes

Within the tags used in the body of the text there are a number of attributes that can be used. Attributes are added into the tags that add an additional command or declaration to the command. *Figure C* lists possible attributes, and their added command. Attributes may not be added to all commands, only select attributes correspond with select tags.

FIGURE C

ATTRIBUTES	
TAG	**FUNCTION**
<...align = (left/right/center/justify)>	changes alignment of text or image
<...width/height = (# 1-100)%>	changes size of image to a percentage of the viewing screen
<...height/width = #>	changes the image size to a select number of pixels
<... href = " file ">	declares reference address
<...name = " ">	marks part of the document for reference
<...bgcolor = "# or name">	alters background color from defualt gray
<...border = #>	created a border - used with image command or table command
<...scr = "file">	states address, used with images and links

3. D. 2. Tables, lists and frames

To better organize information it is possible to create tables and lists with HTML. Frames also help organize data and can be aesthetically pleasing.

FIGURE D

TABLES	
<table> </table>	declares table
<table border = #>	defines and assigns table border thickness
<th> </th>	defines table heading
<tr> </tr>	defines table row
<colspan = #>	changes the # of columns the cell covers
<rowspan = #>	changes the # of rows the cell covers
<nowrap>	keeps data in one line
<td> </td>	defines table data

LIST	
<dl> </dl>	declares description list
<dt> </dt>	definesdata term
<dd> </dd>	defines data description
	declares unnumbered list
	defines item in list
	declares a numbered list
<menu> </menu.>	declares a menu
<dir> </dir>	declares a directory list
FRAMES	
<frameset> </frameset>	declares frames **cannot be used inside the body tag**
<frame> </frame>	
<...cols = "value %, value%">	assigns the columns to a percentage of the page
<...cols = "value, value">	assigns the columns to pixel values
<...cols = " * , value">	assigns the second column a fixed value and designates the rest of the screen to the first column

3. D. 3 Links

Probably the most important feature of any HTML document is connecting it with other pages. This is accomplished through links.

FIGURE E

LINKS	
 	delcares anchor --reference
 	declares anchor --name
<...a link = color >	alters default color of an active link
<... vlink = color>	alters default color of a visited link
<... link = color >	alters default color of a link

To see an example of links, tables, and lists please refer to *appendix A* and *appendix B.*

4. SUMMARY

HTML is a computer language used to create Web pages for the Internet. An HTML file is created on a word processing program, and is viewed graphically through the Internet. The HTML file consists of tag commands that create a document known as a Web page. The basic structure of an HTML file consists of the head and body. Within this structure exists a plethora of possible tags to better organize data, create style and link information with other relevant sources.

```
<html>
<head>
<title> Kristine Angell's demo page </title>

        <!-- Please note that some commands that appear on netscape do
        not print out onto paper ex. <blink> command and any color.
        To see the results of this html file copy this page into a word
        processing file and open through nestscape --->

</head>

<body bgcolor = "ffff33" alink = white vlink = red link = black>

        <!-- ** note ** the bgcolor is a number from the RGB color values and
        can be found at http://www.iconbazaar.com/216color.html
        Basic colors can be named as the links are. -->

<font size = 4 color = green>
<center>
<h1>DEMO PAGE</h1>
</center>


 <img src = "/home/angellk/AIORWEB/y.gif">
 <a href = "#a">List 1 - this type of link jumps to another section of the page
 by naming a section of the page -- Jump to tables</a><br>
<img src = "/home/angellk/AIORWEB/y.gif">
 <a href = "#b"> List 2 - unordered</a><br>

 <hr>

I will attempt to place most commands listed into this page for demonstrational
purposes.<br>
<blockquote> block quotes are one of the few ways to indent text.<br>
        "Choose life.  Choose a job.  Choose a career.  Choose a family.
        Choose a big television, choose washing machines, cars, compact disc
        players, and electrical tin openers.
        <br>Choose good health, low cholesgerol, and dental insurance.
        Choose fixed-interest mortgage repayments.  Choose a starter home.
        Choose your friends." <cite> --Trainspotting</cite>
        </blockquote>

The commands for
<b>tables </b>
and
<b>lists </b>
produce these results:
<p>
<center>
<a name = "a"><table border = 6></a>
        <tr>
        <th colspan = 7><font size = 5>Weeks</th>
        </tr>

        <tr>
        <th>MONDAY</th>
        <th>TUESDAY</th>
```

```
      <th>WEDNESDAY</th>
      <th>THURSDAY</th>
      <th>FRIDAY</th>
      <th>SATURDAY</th>
      <th>SUNDAY</th>
      </tr>

      <tr>
      <td>Virginia</td>
      <td>Long Island</td>
      <td><fotn color = blue>Cortland</td>
      <td>Cornell</td>
      <td>Maryland</td>
      <td>Taberg</td>
      <td>Adorondacks</td>
      </tr>

      <tr>
      <td>Long Island</td>
      <td><font color = green>Cortland</td>
      <td>Cornell</td>
      <td>Maryland</td>
      <td>Taberg</td>
      <td><font color = green>Adorondacks</td>
      <td>Virginia</td>
      </tr>

      <tr>
      <td>Maryland</td>
      <td rowspan = 3>Virginia</td>
      <td><font color = blue>Cortland</td>
      <td>Taberg</td>
      <td><font color = "cc66ff">Adorondacks</td>
      <td>Long Island</td>
      <td>Cornell</td>
      </tr>

      <tr>
      <td>Long Island</td>
      <td><font color = red>Cortland</td>
      <td>Cornell</td>
      <td>Maryland</td>
      <td>Taberg</td>
      <td>Adorondacks</td>
      </tr>

      <tr>
      <td>Long Island</td>
      <td>Taberg</td>
      <td>Cortland </td>
      <td>Cornell</td>
      <td>Maryland</td>
      <td>Adorondacks</td>
      </tr>
/table></center>

      <!-- To change the font size and color in a table from the defualt
      settings, size and color must be specified for each element in the
```

```
        table. -->
<p>
The first list mimics a
<u>glossary</u>
 with word/defenition style. The second
list is <i>unnumbered</i>, and the third list is <blink>numbered.</blink>

<dl>
<dt>Cortland</dt>
<dd>the <font color = red> dragons</dd>
<p>
<dt><font color = blue>Cornell</dt>
<dd>the Bears</dd>
<br><br>                <!-- two '<br>' are the same as one '<p>'--->
<dt>Long Island</dt>    <center>        12-11</center>
<dd><s>New York</s></dd>
</dl>

<font color = green>
<img src = "/home/angellk/AIORWEB/sm.gif" align = right width = 6% height = 9%>

<a name = "b"><ul></a>
<li>Carl</li>
<li type = circle>Mark</li>
<li type = square><font color = red>Chuck</li>
</ul>

<a href = "http://www.iconbazaar.com">
<img src = "/home/angellk/AIORWEB/sm.gif" align = right></a>
<font color = green>

<ol>
<li>Rachmaninoff</li>
<li>Mozart</li>
<li>Coppland</li>
<li>Chopin</li>
</ol>

<pre>"Measuring a summers day,
        only find it slips away. . . to gray. . .
Thinking how it used to be.  Does she still remember times like these.
To think of us again, and I do."  Led Zeppelin

        Here I can place  more    than     one    space!!!!  And Tabs.</pre>
<hr>
<address>this web page was created by Kristine Angell</address>

</body>
</html>
```

APPENDIX B

DEMO PAGE

* List 1 - this type of link jumps to another section of the page by naming a section of the page -- Jump to tables
* List 2 - unordered

I will attempt to place most commands listed into this page for demonstrational purposes.

> block quotes are one of the few ways to indent text.
> "Choose life. Choose a job. Choose a career. Choose a family. Choose a big television, choose washing machines, cars, compact disc players, and electrical tin openers.
> Choose good health, low cholesgerol, and dental insurance. Choose fixed-interest mortgage repayments. Choose a starter home. Choose your friends." --*Trainspotting*

The commands for **tables** and **lists** produce these results:

Weeks						
MONDAY	**TUESDAY**	**WEDNESDAY**	**THURSDAY**	**FRIDAY**	**SATURDAY**	**SUNDAY**
Virginia	Long Island	Cortland	Cornell	Maryland	Taberg	Adorondacks
Long Island	Cortland	Cornell	Maryland	Taberg	Adorondacks	Virginia
Maryland	Virginia	Cortland	Taberg	Adorondacks	Long Island	Cornell
Long Island		Cortland	Cornell	Maryland	Taberg	Adorondacks
Long Island		Taberg	Cortland	Cornell	Maryland	Adorondacks

The first list mimics a glossary with word/defenition style. The second list is *unnumbered*, and the third list is numbered.

Cortland
: the dragons

Cornell
: the Bears

Long Island

New York

- Carl
- Mark
- Chuck

1. Rachmaninoff
2. Mozart
3. Coppland
4. Chopin


```
"Measuring a summers day,
        only find it slips away. . . to gray. . .
Thinking how it used to be.  Does she still remember times like these.
To think of us again, and I do."  Led Zeppelin

        Here I can place  more    than     one    space!!!!  And Tabs.
```

---

this web page was created by Kristine Angell

BIBLIOGRAPHY

BOOKS

Pfaffenberger, Brian; WORLD WIDE WEB BIBLE, Mis:press 1995

Kehoe, Brendon P.; ZEN AND THE ART OF THE INTERNET, Prentice Hall PTR, 1996

Maurer, Herman; Kappe, Frank; HYPERWAVE. THE NEXT GENERATION WEB SOLUTION, Addison Welsey Longman, 1996

Mara, Mary Jane; THE MAC GUIDE TO THE WORLD WIDE WEB, peachpit press 1995

WORLD WIDE WEB

Hannah, Micheal J.; Sandia National Laboratories : Mjhanna@sandia.gov

Quad, O.J.: http://www.quadzilla.com/reference/reference.htm

Grobe, Michael; The University of Kansas: grobe @kulub.cc.ukans.ed

WRITING WORLD-WIDE WEB (WWW) PAGES

Stefan Enjem

Whitesboro High School
Whitesboro, NY 13492

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

September 1996

Writing World-Wide Web (WWW) Pages

Stefan Enjem

Abstract

This paper documents a Rome Laboratory summer program to research, design, implement and populate a set of Web Pages for the Intelligence Technology Branch (RL/IRAE) World Wide Web (WWW) site. It describes the design considerations and alternatives involved in creating such a site, an assessment of existing tools and utilities deemed useful for Web page authors and the implementation of RL/IRAE's Web pages. With its recommended design approach, its lessons learned, and the inclusion of pointers to other valuable sources of information, this report should serve as a useful guideline to beginning World Wide Web page authors.

Writing World-Wide Web (WWW) Pages

Denise Doyle	*Jim DiIorio*	*Al Sisti*	*Stefan Enjem*
Rome School System	*Madison Oneida BOCES*	*Rome Laboratory*	*Whitesboro High School*
Rome, NY 13440	*Verona, NY 13478*	*Rome, NY 13440*	*Whitesboro, NY 13492*

Note: This report was made exclusively for a WWW Browser. Instead of referring to a hard copy please visit Http://www.rl.af.mil/Lab/IR/IRA/IRAE/Pubs/WWW_Paper/WWW_Paper.html. Thank you.

INDEX

INTRODUCTION:

The Internet: Background and Concepts

Twenty-four years ago a need for high-speed transfer of information existed in the field of research. The Internet was created in 1973 as an answer to this need and as a tool to aid the United States Defense Advanced Research Projects Agency (DARPA) in communication between computers. Over the next ten to fifteen years,

using the Internet required an extensive computer background and working knowledge of computer systems and computer programming.

As time progresses, more and more people are becoming familiar with the Internet. And yet, there is really no agreed upon definition for the word Internet. Basically, it is an enormous network of networks based upon TCP/IP protocols. The Internet allows people in the fields of education, research, government and business to collaborate quickly and easily through e-mail, discussion groups and conferencing. This network enables its millions of users around the world to contact others in their fields to discuss topics of interest and exchange ideas as well as do research.

One of the greatest features of the Internet is the availability of a pool of free programs, sometimes referred to as "shareware." Downloading this software enables users to experiment with techniques that ordinarily would not be as feasible.

There are a variety of methods and tools to access the vast amount of information available on this "Information Superhighway." Some of the more prevalent tools are:

* ftp - Ftp is an acronym for File Transfer Protocol. It is the standard method of transferring files between remote computers. Normally a user should use the word "anonymous" for the user name and his/her e-mail address when logging onto an ftp site.

* telnet - Telnet is used when remotely logging onto one computer from another.

* gopher - Gopher is a menu-based system for searching Internet resources.

* archie - Archie is used for locating computer files on the Net that can be downloaded via ftp. It contains a database with directories of what is available via ftp.

* veronica - This is a search tool used with gophers. Veronica is to gopher what archie is to ftp.

* WAIS - Wide Area Information Server - This is a system for retrieving information in databases and libraries.

* WWW - World Wide Web - The World Wide Web is one of the fastest growing networks within the Internet. Phrases within WWW documents contain hypertext links which allow one piece of text to cross-reference other pieces of text within the same document or in another document; either on the same server or another.

The World Wide Web (WWW)

The growing popularity of desktop computers in the late 1980s led to the need for a more user friendly means of communicating over the Internet. Researchers wishing to communicate and share information with colleagues did not always have the expertise to use the Internet. This need led to the creation of the World Wide Web. The Web was created and developed into the application that exists today, between the years 1989 and 1991. The European Center for Nuclear Research (CERN) in Geneva, Switzerland is credited with much of the creation and development of the World Wide Web.

For more information about the history and creation of the World Wide Web, the reader may wish to refer to one of the following WWW sites:

"About The World Wide Web"; McLaurin, Alistair; The Internet Society;

"WWW Frequently Asked Questions (FAQ)";

One of the most exciting pastimes involving the World Wide Web is navigating (also called "**surfing**") through various sites and home pages. **Home pages** are base sites for a variety of different businesses, schools, retailers, and individuals. Some home pages employ very unique and creative styles; incorporating different degrees of graphic images, pictures, background patterns, and multimedia clips. One other point bears noting here, and that is the distinction between a Home Page and a **Start-up Page**. These terms are often used synonymously, although there are subtle differences that make such interchanges incorrect. A Home Page is a document that should describe a site or person to the rest of the world, while a Start-up Page is a page a user creates that will be the first page viewed when the user's browser is started. It should contain a person's favorite bookmarks or sites often visited. It should also be kept as small as possible so as to start up quickly when the browser is opened. When using Netscape, the Start-up Page is also the page the browser goes to when the user selects the Home button (thus the confusion).

The standard language used for communicating between WWW clients and servers is known as Hypertext Transfer Protocol, or **HTTP**. It is also the prefix of a site's Internet address, as will be described later. When Web pages are displayed they are text files that have been adapted to a uniform markup language called Hypertext Markup Language (**HTML**). Other markup languages exist throughout the Internet (e.g. HotMetal, VRML); however, HTML is the most prevalent. Converting text documents to HTML requires the configuration of HTML styles and tags (described in more detail later in this paper) which affect how a document and the elements that make up the document are displayed. Hypertext Markup Language is simple and easy to understand, and has software and software extensions that are currently available to convert common text file into basic HTML documents. For more information on HTML refer to:

"A Beginner's Guide to HTML",

Before the Web can be accessed, a program has to be chosen as a navigation tool, called a **browser**. Browsers control how documents are displayed and also control the extent to which more advanced features (e.g. multimedia, audio) function. The two most common browser programs are Internet Explorer and Netscape Navigator. After careful consideration, Netscape was selected for the work done for this paper.

The Web is unique in the sense that it catalogs information by subject. Most WWW work sessions begin with a search, using a program called a **search engine**. For more information on search engines refer to "Search and Research: A Study of Internet Search Engines and Techniques".

Regardless of the search engine used, the result of the specific search is an ordered list of WWW sites containing the search qualifier.

Navigation throughout the Web is done by simply clicking on key words or phrases, called **Hyperlinks**. Tags within an HTML document are identified as Hyperlinks and anchor tags connect these links to other identified links in the same document or other HTML documents. Finally, Web sites are identified by "addresses" called Uniform Resource Locators (**URLs**), and all documents, links, graphics, pictures and elements, etc., have URLs. These URLs are a coded syntax identifying the server source locations, access methods and computer locations; where the first prefix is the Internet navigational method being used. A country code is often included in the URL as well.

With the basic definitions now out of the way, it is now possible to cogently describe the general guidelines that should be followed in building the pages of a WWW site, and the design alternatives that are considered within that process.

"Where Do I Begin ...."

Building a WWW page can be as simple as filling in a pre-formatted template, or as detailed as a built-from-scratch page, based on an in-depth knowledge of markup languages (such as HTML), graphic design, prevalent file formats and the potential target audience. Each approach has its advantages and disadvantages. Too simple a page, while minimizing (or in some new applications, eliminating) the need to

know HTML, may be so bland as to discourage return visitors; while a page so busy with multimedia clips and replete with hyperlinks may actually detract from its intent; that of information presentation.

More often than not, however, Web page authors are becoming more and more creative in coming up with innovative pages, in order to set themselves apart from others competing for the favor of an ever-increasing market of "data miners", shoppers and general "surfers". As expected, this requires a technical knowledge of page construction, an understanding of the expectations of the target audience and familiarity with the state-of-the-art of innovative pages. (Netscape Communication Corporation's Home Page facilitates this familiarity by maintaining two dynamic lists: "What's New?", which is devoted to new sites coming on-line to the Internet daily , and "What's Cool?", which is a list of sites that have caught the eye of the Netscape "cool team". Another surfing tool ideal for viewing innovative and creative sites is the Netscape "Galleria" which is a showcase of innovative customers who have created innovative sites using Netscape Server software. For more about the Netscape galleria and Netscape Server software, the reader is referred to:

http://home.netscape.com/escapes/galleria.html)

.

Learning HTML

Once the decision is made to eschew the use of templates, or automatically-generated home pages, the Web page author will need to learn and understand HTML; whether he/she decides to adapt an existing page or build one from scratch. There are some excellent books already in print on the subject, as well as some useful sites on the WWW. As a starting point, the most helpful document is:

"A Beginner's Guide to HTML",

This guide introduces the user to the basic HTML tags, linking to documents, and the displaying of images and sounds. After reading this document, it is suggested that the new user go through one of the following tutorials when writing his/her initial Web page:

"Writing HTML",

"Wade's HTML Tutorial",

After the user is familiar with the HTML language, it is essential to have a copy of the following guide to use for reference when writing future pages:

"Bare Bones Guide to HTML",

The Bare Bones Guide is a site maintained by Kevin Werbach. It does not contain a detailed explanation of HTML; rather, it is a guide for users already familiar with the language.

An HTML document is nothing more than a basic text file. In fact, the more basic it is, the easier it will be to work with. All styling of the document (such as type sizes, alignment, spacing, special characters; to name a few) occurs as a result of the addition and placement of HTML tags. Therefore, the more basic word processing applications, such as TeachText or SimpleText, may be used to actually construct the HTML document. Other more advanced word processing programs (e.g. Word or MacWrite) should have an option under the "Save as . . ." menu such as "text" or "ASCII"; this is how the HTML document should be saved or formatted. MicroSoft Word for Windows even has an option to "Save As . . ." an HTML document; others are certain to follow. A more detailed description of some of the tags and the format of an HTML document is given later in the paper.

Adapting an existing Web page

A common practice of beginning Web page authors is to locate a page/site on the Web, and to modify it to meet the requirements of his/her own page. (A discussion of the ramifications of "ownership" of a Web page is well beyond the scope of this paper). Once identified, the page may be printed as it is viewed in Netscape (choosing "Print . . ." from the file menu), or the page can be viewed and printed as a source document. Saving and printing a source document allows the user to view the page as an unformatted HTML document; i.e., not as it appears via Netscape. Each element within the document is broken down into the specific tags that generate the look, style and appeal of the page display.

Web Page Design Considerations

As mentioned earlier, the intent of a Web page is to present information to an ultimate consumer of that information. In doing so, there is a push-pull aspect that drives the design of the page, inasmuch as it is meant to reach (or more accurately, to be reached by) a wide audience, whose processing and display capabilities are unknown. This implies the need for judicious selection of style guides, file formats, browser, markup language, search engine and conversion utilities; so as to find the 'least common denominator' that allows access from the greatest cross-section of the WWW community. There are several search engines, browsers and markup languages that were (or could have been) considered as part of the WWW page construction tool kit. As an example, Hypertext Markup Language (HTML) is very popular and used extensively with Netscape, however, some consideration must be given to other markup languages and other browsers that may be used by potential viewers. Some elements of the home page may not be viewable or accessible when using other markup languages or other browsers. Sometimes it is possible, however, to include links to free shareware applications or utilities (e.g., graphic, audio and/or multimedia programs) that may be downloaded by viewers for the proper display and function of the home page.

The ease with which a page is accessed or traversed needs also to be considered. Although detailed graphic images and pictures enhance the aesthetic quality of a document, one must also consider the loading time involved in displaying such images. A person surfing the Web will often abort the loading of a document that takes longer than one or two minutes. Also some potential viewers of the document may not possess the computer hardware required to display large graphic files, as well as any audio and/or multimedia files.

The degree of community acceptance of WWW tools, formats and access methods is a major driving factor in the use and evolution of a user's Web page. Similarly, the extent to which these tools and standards are maintained and supported should play a vital role in the design approach. Again, if information cannot be downloaded or, once downloaded, displayed, it is useless to the consumer; and ultimately, so is the Web page.

Finally, the internals of the page/site must also be considered; that is to say, the actual information content. For example, during the design and construction of the Rome Lab IRAE home page (described later in this report), a variety of considerations had to be addressed and a list of design criteria was established. This design criteria is important to ensure that the proper audience is reached, and that the design is financially and technically feasible. The first consideration was to determine the degree of support from the personnel of the organization. An informal poll of sorts was conducted to develop a list of elements that were to be included in the document, and whether unanimous or majority support was to govern the inclusion or deletion of each element within the document. The document must accurately reflect the people and the organization it represents. Additionally, the document must be morally and professionally acceptable to the targeted audience. It is very important that all parties involved in the organization feel comfortable with the structure and content of the home page before any extensive work gets underway.

WWW Page Construction

The Web page construction process involves a great deal of trial and error, including many iterations of displaying the page in Netscape to test the appearance and operation of the document. This may be accomplished by simply opening Netscape, choosing "Open file" from the "File" menu and selecting the home

page HTML document. This process will display the document exactly as it will appear on the World Wide Web.

The experimentation (trial and error) stage involves making small modifications within the HTML document, then switching to Netscape, reloading and viewing the results of the modifications. As an aside, the authors found that this process is much easier when the computer hardware being used has sufficient Random Access Memory (RAM) to allow both Netscape and the text application used to construct the HTML document (e.g. MicroSoft Word or TeachText) to be opened simultaneously.

The modifications to the document will probably involve experimenting with different type styles (such as bold or italic), different sizes, alignment adjustments, etc. Additionally, adjustments to the alignment of graphic images, pictures and/or other lines or small icons will be required. Probably the most tedious modifications will be those dealing with the adjustments and/or placement of the Hypertext tags.

Hypertext tags, as mentioned earlier, generate the look, style and appeal of the page display. Often these tags must be modified, relocated, rewritten, or be totally replaced by different tags, until the desired look or function is achieved. This process is time-consuming, because Netscape gives no clue as to why certain tags don't function the way they were designed to. Trouble shooting a document is totally trial and error.

Manipulating and Converting HTML Elements

Elements that make up a Web page include a number of different file types such as graphic images, pictures, multimedia clips, audio clips, various colors and background patterns. When importing, searching for and/or creating different files to be used in Web pages, various converters are available to aid in proper formatting.

Most often text is generated using a word processing program. It is possible, however, to import text files that have been created from other sources. This importing process is easiest when the text file is basic or plain text. If the file is anything other than basic text, the file must be converted to basic text. This may be accomplished by opening the file using the application with which it was created and saving it as text, or ASCII, etc. The problem may arise, however, that a file is not a text only file, and the application that was used to create the file is unavailable. Conversion applications are available that can import, open or convert word processing files. Some of these applications are available as shareware and may be found on the Web.

Adding graphic images to a document greatly enhances the aesthetic quality of the document. The most common graphic file format (and the one most readily supported by Netscape) is Graphic Interchange Format (GIF). If a graphic file is not a GIF file, it must be converted. This process is made easy using a GIF converter, which is shareware and available on the Web. For more information, the reader is referred to:

http://www.kamit.com/gifconverter.html

Other graphic programs, such as Adobe PhotoShop or Aldus DarkRoom, enable the user to open many types of graphic files and resave them in other formats, including GIF. These programs also allow users to retouch, resize and otherwise manipulate the images; thus opening many creative avenues.

Once a graphic file is stored in the document location and placed in the Web page, manipulating the image is not difficult. After using a program such as PhotoShop to alter the image, simply saving the file before closing the program will replace the existing file.

All graphic files are rectangular in shape and have an associated background color; which will not always match the background of the Web page. The image background may be hidden by making the background color transparent. This is achieved by using a program called "Transparent". This program is not available as shareware and the authors of this paper were unable to find any similar program as shareware, however, the program is relatively inexpensive. As the name implies, this program will create transparent colors. The drawback to this approach is that "Transparent" will only allow the user to select one color to be transparent.

Graphic images may be used as backgrounds of Web pages, by using the HTML "background" tag. The image may be repeated to achieve a pattern effect or the image may be displayed in its entirety, occupying the entire background of the Web page. It may be necessary to experiment with the manipulation of the original file until the desired results are achieved.

There is an array of HTML tags that result in different graphic effects, as well as different display and alignment options. Some study and experimentation may be necessary to achieve the desired graphic results. If the idea of a graphic effect is generated from another Web page, it is possible to copy the image itself or the HTML tag that controls the effect. When a Web page is open, saving an image is achieved by simply clicking and holding on the image(or right-clicking on a pc). This will result in the appearance of a pop-up menu offering a variety of options, one of which is "Save Image As...".

A Web page author will find it beneficial to experiment with the tags. For example, how does the document change if a tag is moved to a different position in the document?

Testing and Finalizing The Web Page

Finally, when a Web author arrives at the stage when the Web pages are finished, it is necessary to test the pages through other browsers. This will allow the author to see how the document will be viewed via other browsers such as Internet Explorer. Testing the pages in this fashion will enable the author to ensure that his/her site is accessible to the largest possible audience. It will more than likely be necessary to test and modify the pages several times until the desired results are achieved. There are also anchor checkers available that will ensure that the Hypertext links are properly attached to a location. Anchor checkers should be used not only during the testing phase, but also periodically after coming "on line" in order to avoid linking to addresses that have changed.

IRAE Branch Pages: The Concepts are Applied

(The appendix in this paper contain the authors' Rome Laboratory IRAE WWW pages. These pages may be used as templates for other Rome Lab organizations wishing to create Web pages).

TABLES

On the IRAE Home Page (Figure-1) a table was made. It is made up of four columns and six rows. We also added pictures to illustrate where the user is going to go if he/she clicks on that particular part of the table. This table makes navigating our IRAE site much faster and more efficient. The table was made using various HTML tags. These tags include the TD and TR tags. The TR is for a Table Row. The TD is for Table Data. A table is started with the TABLE tag. The TR tag contains all the TD tags. There can be multiple TD tags. To view the source for the table below go to the view pull-down menu, and click on View Source. Our Modeling and Simulation page is also set up the same way. Click on the Modeling and Simulation part of the table to view that page.

<TABLE>

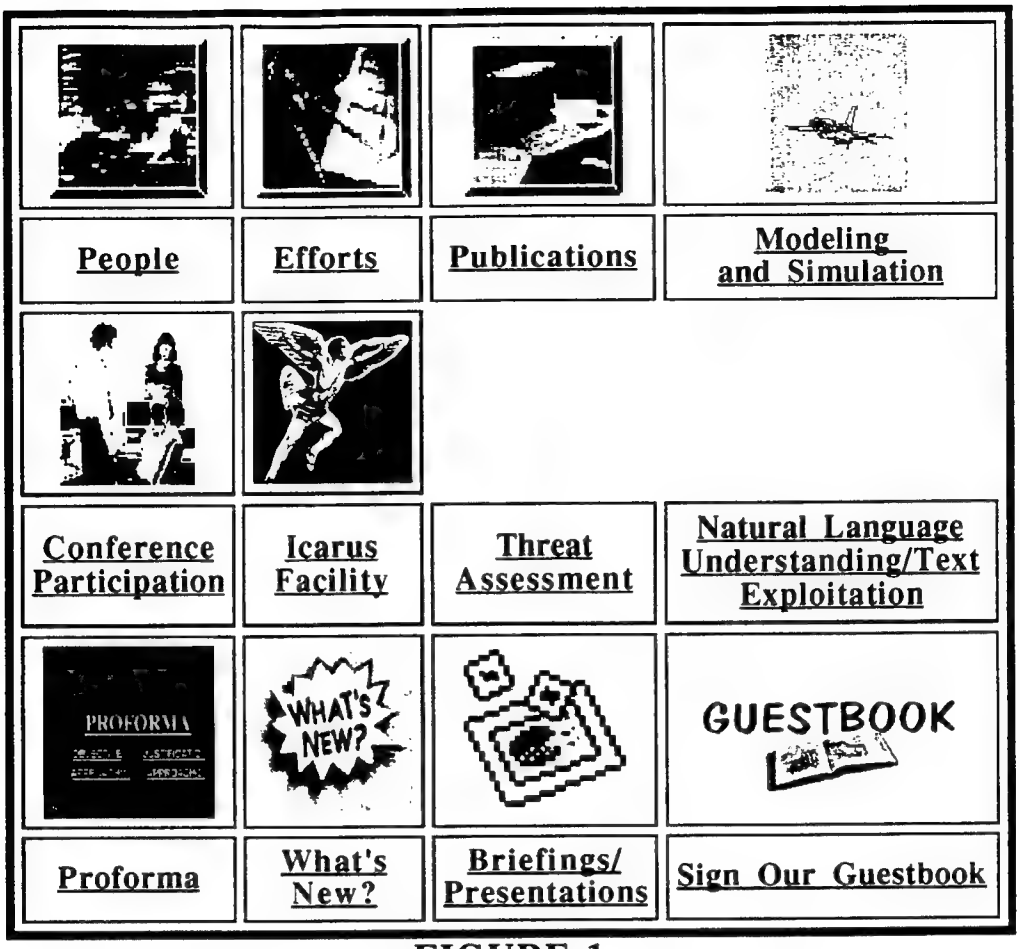


FIGURE-1

</TABLE>

Our Publications page is made up of most of the publications people at IRAE have composed. They are also set up in table format, much like our home page. We have added access icons to make choosing a desired format as easy as clicking. We are currently working on making more file formats available to our users. If they are not on the site we try to make sure that there is a "mail to" icon. Below is an example of the publications page (FIGURE-2).

Sisti, A.F. and Trott, K.C., "A Distributed Interactive Simulation (DIS) Environment for C3I Capability Assessment", 5th IEEE Dual Use Technologies and Applications Conference, May 95	
Farr, S.D. and Sisti, A.F., "Visualization of General-Purpose Simulation Results", 4th IEEE Dual Use Technologies and Applications Conference, May 94	
Tew, J.D.,PhD, Sargent, R.G., PhD, Zeimer, M.A., and Sisti, A.F., "Metamodel Procedures for Air Engagement Simulation Models", RL-TR-93-202	
Sisti, A.F., "Electronic Combat Visualization Demonstration (ECDEMO)", RADC-TM-90-4	
Farr, S.D. and Sisti, A.F., "Fortran-Callable Ramtek Interface Program (FCRIP)", RADC-TM-89-5	

FIGURE-2

SLIDES

On the Enabling Modeling and Simulation Technologies Page we made an index of the slides for this presentation. Also on the index page is a downloadable version. This is for people who don't have time to view the presentation online. Each slide has buttons to move either forward or backward in the presentation. Below is an example of a section of the index, along with an example of a slide(FIGURE-3).

- **Downloadable Version of Presentation**
- **Readable Version of Presentation**

Slide 1.Enabling M&S Technologies
Slide 2.Presentation Outline
Slide 3.Enabling Technology for Real-Time Simulation
Slide 4.The Air Force Hierarchy of Models Concept

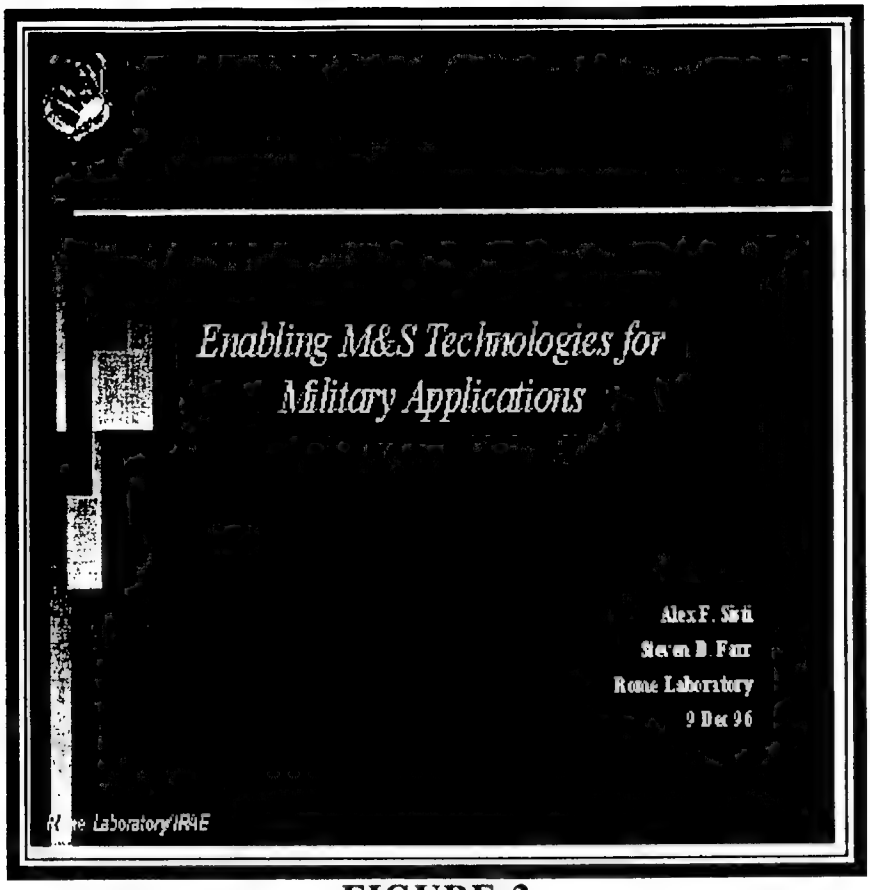


FIGURE-3

IMAGE MAPS(example map)

Some Web sites can get quite large. One navigation alternative is to make an image map. An image map is just that; a map made by an image. There are three parts to an image map. First, the actual image. Second, the .map file, which holds the coordinates. The third part is the html that tells the browser to use the .map file with the image.

There are three shape types to make hot spots on your map. Hot spots refer to places on the map that link to other HTML documents. The three main types of shapes are a rectangle, a circle, or a polygon. The shape we used here at IRAE was the rectangle.
Rectangle: There are two coordinates that need to be defined; those being the upper-left and lower-right corners of the location of the hot spot. To find the coordinates, open the image in a graphics viewer that has coordinate capabilities.
Once you have your image, it is time to make the .map file. Name this file as follows:
imagefile.extension.map; for example, irae.gif.map.
After you have named your file, set it up like this:

default http://your_path/ /* This sets the default places on the map. For example, if you click somewhere on the map that is not a hot spot, it will take you to your default page. It may be convenient to make it the page where your map is located. */

```
<A  HREF=/your_path/image.gif.map><IMG  SRC=/your_path/image.gif  ISMAP></a>
```

ADDING META TAGS

Some documents on a Web site can get extensively long. In this case, you may want to add meta tags. Meta tags are anchors that take you to certain spots in a document. You set them by putting

```
<A  NAME=name_of_meta_tag>
```

where you want the link to anchor to. Then close the tag with a

```
</A>
```

After you have set the meta tag, you must set the link to it. Probably the most convenient place for the link would be at the top of the document. If you want you could put it on a totally different page. Set the link up like this:

```
<A  HREF=http://your_path/your_page.html#your_meta_tag>
```

To close the link just add

```
</A>
```

A GUESTBOOK is also a nice addition to your site. Four pages must be made for this guestbook. A sign in page, a redo page, a thank you page, and a guestbook page. In order to create a guestbook you must write a cgi-script. Don't be scared of writing a cgi-script just because it sounds complicated. In reality it's quite easy. You need a cgi-bin directory with the cgi-lib.pl in it. Remember to name your cgi-script with a .pl extension. Below is the script that was used for the IRAE Guestbook, which can be copied and modified. Note: Make sure to delete the comments.

```
#!/usr/local/bin/perl
require "./cgi-lib.pl";  /* This is the file that we told you about
earlier */
&ReadParse(*in);
print "Content-type: text/html\n\n";
/* Defines all the different rows */
$thismonth = (January, February, March, April, May, July, August,
September, October, November, December)[(localtime) [4]];  /* Local time
insersts the current time. */
$thisday = (localtime) [3];
$thisyear = (localtime) [5];
$name = $in{'name'};
$e_mail = $in{'e-mail'};
$homepage = $in{'Homepage'};
$business = $in{'business'};
$reference = $in{'reference'};
$comments = $in{'comments'};
$outputfile = "/www/htdocs/Lab/IR/IRA/IRAE/Guestbook/gbookbare.html";
/* This is the file that the guest book is written to. */
$outputstore = "/www/htdocs/Lab/IR/IRA/IRAE/Guestbook/gbooksave.html";
/* Temperary storage file. */
$redopage = "/www/htdocs/Lab/IR/IRA/IRAE/Guestbook/redopage.html";
/* This page is if you have not filled out the appropriate information in
the form. */
$thankyoupage = "/www/htdocs/Lab/IR/IRA/IRAE/Guestbook/thankyoupage.html";
```

```
/* This is the thank you page, after a correctly filled out form. */
&redo;
&addbook;
&writefile;
sub redo {
 if ($name eq "" || $e_mail eq "") { /* If statement that says if the
name or email is left empty, open the redo page */
  open(REDOPAGE,$redopage);
  while () {
    print $_;
  }
  close(REDOPAGE);
  exit;
 }
}
sub addbook {
 open(THANKYOUPAGE,$thankyoupage); /* If the form was filled out correctly,
then this page is opened. */
 while () {
  print $_;
 }
 close(THANKYOUPAGE);
}
sub writefile {
 open(STOREFILE,">$outputstore"); /* Stores the data */
 open(OLDFILE,$outputfile);
 while () {
  print STOREFILE $_;
 }
 close(OLDFILE);
 close(STOREFILE);
 open(NEWFILE,">$outputfile");
print NEWFILE<

/* Actual HTML for Guestbook Page */
<HTML>
<HEAD>
<TITLE>IRAE GUESTBOOK</title>
</head>
<body background="/Lab/IR/IRA/IRAE/Images/bg.gif">
<H1>IRAE GUESTBOOK</H1>
<p>
<hr>
/* The $____ stores whatever was inserted for this row in the form */
<b>Name: </b>$name<br>
<b>Homepage:</b><a href="$homepage">$homepage</a><br>
<b>E-mail:</b><a href="mailto:$e_mail">$e_mail</a><br>
<b>Referred By: </b>$reference<br>
<b>Submitted:</b><i>$thismonth $thisday, 19$thisyear</I><br>
<b>Comments:</b>$comments
/* HTML ends here */

stuff
 open(SAVEFILE,"$outputstore"); /* Reading the file */
 $num = 0;
 while () {
  if ($num >= 7) {
    print NEWFILE $_;
  }
  ++$num;
 }
```

```
 close(SAVEFILE);
 close(NEWFILE);
 unlink($outputstore);
}
```

If you read the above code, instead of skipping over it as most do, you will agree that it is pretty much self explanatory. If you don't agree, go through it again. Sooner or later it should click.

FORMS

One current trend in Web Page construction is the inclusion of some type of form. It is through forms that the user can interact with the Web environment. For example, a user may be asked to type in information on how he/she found the site; much like our IRAE Guestbook (the sign-in page of our guestbook is a form). Below is some of the HTML source code for the IRAE Guestbook form. Comments have been placed to show the usage and purpose of most function tags.

```
<Form Action="/cgi-bin/SignIn.pl" method="POST">
/* The form tag starts the form.  The action command uses the SignIn.pl
(mentioned earlier in this report) to post the information entered in the
form.  The method is set to post, to literally post the information on a
different page. */
<center><H2><u>IRAE Guestbook</u></H2></center><p>
<img src="/Images/IR-BAR.gif">
<p><B>Name:</B> /* Shows up on the page (same for other lines below that
are formatted in this fashion) */
<input name="name" size=30 maxlength=60><br>
/* The "input name" is used to name the fields.  The "size" is for the
size of your input box.  Maxlength is for the maximum length that the
users input can be for that box. (same for other lines below that are
formatted in this fashion) */
<b>E-mail:</b>
<input name="e-mail" size=30 maxlength=60><br>
<b>Homepage:</b>
<input name="Homepage" value="http://" size=30 maxlength=60><br>
<b>Business:</b>
<input name="Business" size=30 maxlength=60><br>
<b>How did you hear about our site?:</b>
<SELECT NAME="reference"> /* "Select name" is for a pull-down menu and
the name of the menu. (Note: the name of the menu does not show on the
page) */
<OPTION>ListServe /* The "options" are different options that can be
selected from the pull-down menu. */
<OPTION>Search Engine
<OPTION>NewsGroup
<OPTION>Word of Mouth
<OPTION>Advertisement/Brochure
<OPTION>Friend
<OPTION selected>Web Surfing</select><br> /* This is the option that is
selected when the page is loaded. */
<b>COMMENTS:</b><br>
<TEXTAREA WRAP=PHYSICAL NAME="comments" COLS=50 ROWS=8></textarea><p>
/* This creates a text area with 50 columns.  The name of the text area is
comments and it has physical text wrapping. */
<input type="submit" value="Sign GuestBook"> /* Creates a button that
submits the form to the cgi-script.  The cgi-script then writes to the
guestbook page. */
<input type="Reset" value="Clear"> /* This creates a button that clears
```

```
the form */
</form> /* This ends the form */
```

This is the end of the form portion of the SignIn.html page.

JAVA?

So what is this Java thing you have been hearing so much about? Java brings interaction and animations to the Web. With a Java-enabled browser, your whole atmosphere on the Web can change.
Java is actually a programming language closely related to C++. It was developed by Sun Microsystems to make "executable content" that can be sent out through networks. Executable content? Sounds interesting, but what does it mean? Executable content is a term used to show the difference between what a Java-enabled browser displays and what a non-Java Browser displays. Warning! Java has the ability to get you involved in an in depth interaction. This could keep you on the internet for a much longer time than normal. If you think you spend too much time surfing now, WATCH OUT!
How secure is Java? Java is made so it cannot read or write to your hard drive. For example, it can't download a virus to your hard drive and run it. Java has a client-side verifier that ensures the code is virus-free before it passes to the interpreter.
To learn more about this fascinating Web language go to Http://www.hotjava.com

---

We conclude this implementation section with some lessons learned to prospective page builders:

* Be imaginative; take advantage of the power of multimedia, and the new dimension hyperlinking brings to a document.

* Don't use nebulous phrases like "click here". A more useful hyperlink may be, for example, "For more information on xxx, the reader is referred to .... "

* Don't use glitz for glitz sake. Sites that used the 'blink' capability enabled by Netscape viewing merely to claim to be "Netscape Enhanced" earned the scorn of the Netscape "Cool Team".

* When possible, during the production stage, working files that comprise a Web page should be stored in the same location (folder when using Macintosh). For example the HTML text file, the Netscape file, any graphic or picture files should be stored in the same folder (perhaps titled Home Page Folder). Care must be taken when choosing a title for each document to avoid replacing documents with the same name, thus losing documents. Macintosh will not allow more than one document to be stored in the same location (folder) having the same name. If attempted, the newer document will replace the older document regardless of the application used to create it. One must also realize that this practice does not apply to elements that are linked to other locations.

* After a Web page is already "online", the page should be monitored and updated frequently. The author should notify others if the site is still in the process of being developed, if it is complete or if it (or any of its elements) has moved.

* Assemble a fairly complete set of conversion utilities. Nothing is more frustrating than acquiring the perfect graphic image or sound clip, only to have it rendered useless because it's in a foreign format.

* Eliminate the Home Page/Start-up Page confusion. Don't use the Home Page for the site as the Start-up Page for the browser.

* Above all, remember: "If at first you don't succeed, try, try again." Writing a Web Page is an exercise in trial-and-error. And error. And error. Be prepared to revisit tutorial sites often, as HTML, style guides and formatting tags (and their interrelationships) change often.

Summary

As the World Wide Web promises to continue its exponential growth, those organizations without Web pages will soon be in the minority. Just as Buffalo, NY thrived in the 1800's, solely as a result of being chosen to be a "node" on the Erie Canal (to the detriment of a similarly-sized nearby town that wasn't), so too will the industries, agencies, schools and retailers that choose to come on-line. Enterprising young graduates are even putting resumes on their own home pages. This report will hopefully serve to provide the reader, and the prospective Web page builder, some guidance to constructing and populating a page that is interesting, informative and extendible.

Bibliography

"A Beginner's Guide to HTML"

"An Overview of the Internet and WWW", XEROX Corp. 1994

"Style Guide for Online Hypertext"

"Wade's HTML Tutorial"

AFOSR SUMMER 1997 INTERNSHIP

Jared Feldman
4 Selden Drive
Rome, NY 13440

Rome Free Academy
500 Turin Street
Rome NY

Final Report for;
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, DC

And

Rome Laboratory

August 1997

In the summer of 1997 I held an internship position at Rome Laboratory at the former Griffis Air Force Base under the guidance of Daniel Burns. I learned many great things during my internship at Rome Labs. Among them are programming, computer chip design, and computer chip failure. I also met many excellent scientists and engineers who taught me a great deal.

When I first got to Rome Laboratory I met Dan Burns, who was to be my mentor for the summer. He worked his interns hard, but did an excellent job. He is a genius of a man; one of the smartest people I have ever met. His knowledge of programming and computer chips is immense. I had hoped to meet people of his caliber when accepting the internship. He did a fine job.

The first task Mr. Burns set me to was programming with Labview, a graphical programming language. I had previous programming experience, as I studied the C language for some time. Labview was very easy to catch onto, as it was based on all the principles I knew from C and all other programming languages, except for a radically new syntax. Labview is the future of all programming languages; instead of writing lines of code in text, the programmer constructs a graphical presentation, using the functions built into Labview. For example, when a "for" loop was previously written in text in other languages, it is shown in a box, or "loop" in Labview. The language is quick to understand for someone who has had previous experience in programming. I would call it one of the easiest languages to learn after learning the fundamentals of programming. Labview will replace C++ and Java much like the graphical user interface, Windows, has replaced the text based DOS as the standard operating system for the personal computer.

The next task assigned to me was becoming familiar with computer chip design and manufacture. I learned about the doping process, as well as etching. I also learned about the materials chips are constructed of, and the delicacies of manufacture. It's a very interesting process.

After having learned about microchip design and programming with Labview, it was time to address the real goal of our project: failure analysis. I was stationed in the failure analysis lab, along with Rocco Messagna, a fellow intern. Rocco's main task was to test the reliability of computer chips, and I occasionally watched what he was doing and tried it myself. Chips will fail if any number of problems exist. For example, a run on a chip might have a "nibble", or a nick, that formed during manufacture, or no run at all where one should be. All contact points would be tested on the chip to determine if a failure is present. The missing run would immediately be detected, but if a nibble occurred testing would not detect it. What will occur later in its life is "burn out", where the run which contains a flaw fails due to heat build up through resistance in the conductor. This typically happens four to seven years down the road, but could happen at any time after the chip has been installed and set into use. This can be a huge setback in the product's use, not to mention the cost involved, if the microchip is in place in an application such as a satellite or the Hubbell space telescope, and fails. A substantial amount of data would be missed, as well as much time and money spent to replace the culprit chip.

With this dilemma in mind, Mr. Burns set out to design a method to test chips for nibbles in the runs. He devised a way to test the chip by splitting the runs and adding more runs and contact points in the manufacturing process, thus having a built in testing

ability. He holds a patent for the idea, as it is the only one in existence. His method will someday be used in all chips, but for the time being it will only be used in chips onboard vehicles and instruments in the space program and other high cost, hard to access priorities. Mr. Burns' method adds slightly to the cost of the chip and roughly a quarter of its length to the overall size of the chip. As time goes on and prices come down, the failure design method will be utilized in nearly all computer chip manufacture.

My part in the improved computer chip project was to update and write a program to split the chips and add more material. Of course, we used Labview, and the program turned out quite nice. By 22 August, when my internship expired, the program ran fine but still needed some minor "tune ups" to make the user's job easier. I really enjoyed seeing something practical come out of my newfound knowledge of Labview and computer chips.

I am very thankful that I was given the opportunity to work at Rome Laboratory. Mr. Burns proved to be an excellent mentor whom I enjoyed very much. I must say again, he's a genius. I also enjoyed the other fine scientists I met over the summer. I learned many things from them, and used the most advanced equipment offered today under their guidance. It was great to learn so much about technology, and I'm proud to say that I had a part in technology's future. I took part in a great program that will hopefully let many more young interns understand more about technology in the years to come, just as I did.

EXAMINATION OF THE NEAREST-NEIGHBOR RULE IN VOICE PATTERN CLASSIFICATION

Douglas M. Feldmann

Oneida High School
560 Seneca Street
Oneida, NY 13421

Final Report for:
High School Apprenticeship Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington DC

and

Rome Laboratory

August 1997

EXAMINATION OF THE NEAREST NEIGHBOR RULE IN VOICE PATTERN CLASSIFICATION

Douglas M. Feldmann
Oneida High School

Abstract

Voice pattern classification is difficult especially if there are two separate voices speaking simultaneously. it is possible to determine classes using the k-nearest neighbor method. There are different methods of determining k-nearest neighbors. This paper with evaluates two of the methods and shows how they can be implemented.

EXAMINATION OF THE NEAREST-NEIGHBOR RULE IN VOICE PATTERN CLASSIFICATION

Douglas M. Feldmann

Introduction

The nearest neighbor rule can be applied to the classification of co-channel speech. However, this method is not exceptionally accurate and many computations are necessary. Accuracy can be increased by extending the process to k-nearest neighbors, but the number of computations reach unreasonably high levels. There are several methods available for simplifying the computational process. Multi-dimensional data can be projected to a single dimension, and the data can be clustered. Then the powerful branch and bound tree search method can be used. The combination of these processes result in fast computation of the nearest neighbor.

Discussion

We will be classifying several different voice patterns. Each data set contains 36 dimensions, with 2000 samples in each. 4 data sets will be used.

1.) Voiced/Voiced - two voiced speakers

2.) Unvoiced/Unvoiced - two unvoiced speakers

3.) Voiced/Unvoiced - 1st speaker is voiced while 2nd is unvoiced

4.) Unvoiced/Voiced - 1st speaker is unvoiced while 2nd is voiced

The data sets will be combined into a 36 by 4000 matrix. Half will be voiced and the other half will be divided equally between the other three categories. All of the data will be tested using programs written for MATLAB High-Performance Numeric Computation and Visualization Software.

Methodology

The first set of tests were done using the simple voting method for finding k-nearest neighbors. The following code takes each observation point and calculates the distance to each data point and finds the closest. The results are then classified using predetermined classification vectors. The user can set the number of neighbors to

find, thus increasing the accuracy of the program. However, each neighbor multiplies the time required to complete the program.

```
.
clear
load knnrn %knnrn is the data file containing data as described above and an observation sample.
halfn=n/2;
totright=0;
k=2;
while (k/2)==round(k/2)
        input('Enter the number of nearest neighbors you wish to find:');
        k=ans;
        if (k/2)==round(k/2)
                disp('**You must specify an odd number**')
        end
end

disp('# of nearest neighbors:');
disp(k)


class=[zeros(1,halfn) ones(1,halfn)];
obs_class=[zeros(1,numobs)];

right_class=[zeros(1,numobs/2) ones(1,numobs/2)];
nn=zeros(numobs,k);
timestart=cputime;
for j = 1:numobs,
  activelist=[zeros(1,n)];
  for s = 1:k,
        B=100000;
        for i = 1:length(data),
                if activelist(i) == 0
                        if sqrt((data(i)-obs(j))^2) < B
                          nn(j,s)=i;B=sqrt((data(i)-obs(j))^2);
                        end
                end
        end %for i
        activelist(nn(j,s))=1;
  end %for s
end %for j

elapsedtime = cputime-timestart;
for j1=1:numobs,
  zero_class=0;
  one_class=0;
  for s1=1:k,
        tmp=nn(j1,s1);
        if class(tmp)==0
                zero_class=zero_class+1;
        else
                one_class=one_class+1;
        end
```

```
 end
 if zero_class>one_class
        obs_class(j1)=0;
 else
        obs_class(j1)=1;
 end
end
for j1=1:numobs
 if obs_class(j1)==right_class(j1)
        totright=totright+1;
 end
end
% disp(nn)
[obs' obs_class']
disp('# correct:')
disp(totright)
disp('Elapsed time:')
disp(elapsedtime)
```

This method is computationally intensive. For example, this process took over 6 min on 66mhz processor, finding only the first nearest neighbor.

The fact that there are 36 dimensions greatly increases the number of computations. To solve this problem we used Fishers Linear Discriminant as presented in Duda and Hart{1}. The program written finds discriminant w that gives the greatest degree of separation of data sets.

```
%The Fisher Linear Discriminant Code
clear
load knnrn
m1=mean(data1')';
m2=mean(data2')';
M1=m1*ones(1,halfn);
M2=m2*ones(1,halfn);
X1=data1-M1;
S1=X1*X1';
X2=data2-M2;
S2=X2*X2';
SW=S1+S2;
SWI=inv(SW);
w=SWI*(m1-m2);
data=w'*data;
obs=w'*obs;
ndm=1
save oned data obs w ndm
```

After w is found, it is multiplied to the data and observation sets. The resulting data is 1 by 4000 and the observation is 1 by 176.

The clustering procedure allows the data to be used in the Branch and Bound tree search method. The program used was based on the k-means procedure found at

http://www-engr.sjsu.edu/electeng/faculty/knapp/HCIRDSFC/C/k_means.htm

The clustering procedure breaks the data down into smaller groups. The program marks each data point as belonging to a specific spot in the tree at each level. These markers are stored as three digit integers.

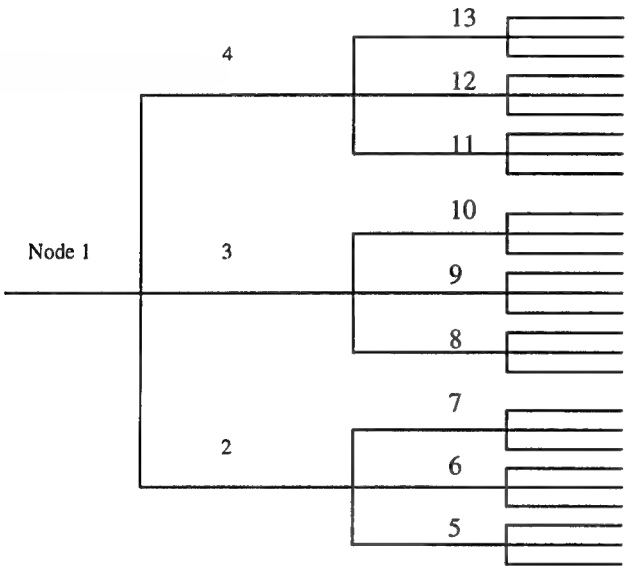


Fig. 1. Shows the cluster tree structure.

Nodes are numbered as shown above, and continued until 40. Data is marked in three levels. Each node can be referenced using the 3 digit code. Each digit is a 1, 2, or 3 representing the branch in that level. The Branch and Bound search requires that the clustering program find the mean of each group as well as each radius. The radius is the maximum distance from any point in the node to the mean of the node.

```
%  k-means clustering
%
%    The k-means clustering algorithm takes a one dimensional data set
%  and separates it into clusters.
clear;
load oned;
%oned Contains data that has been projected to 1 dimension with
%fisher (linear discriminant).
timestart=cputime;

n=length(data);
nummeans=3; % Means in a node after it is broken down
levels=3; % Times the nodes are broken down
branches=3; % Branches in a node

grpmarker=[zeros(1,n)] + 100;

% Matrix containing values for every data point at every level
```

```
% Calculate the total number of nodes
totnodes=1;
for level = 1:levels
      totnodes=(totnodes+branches^level);
end

%   This vector is used to determine which branch from level 1 the current
% level 2 beanch stems from.
findhigherbranch = [1 1 1 2 2 2 3 3 3];
% this data set could be used if more than three levels are needed
% [11 11 11 12 12 12 13 13 13 21 21 21 22 22 22 23 23 23 31 31 31 32 32 32 33
33 33];

m=zeros(nummeans,totnodes); %Matrix for all the means

currentlist=data; %for level 0, currentlist will be the entire data set

meanpos=1; %initialize mean position

%  Main routine

for level=0:(levels-1),
  branchesinlev=branches^level;
  for branch=1:branchesinlev;
    clear currentlist;

    meanrow=rem(branch,3);
    if meanrow==0
      meanrow=3;
    end
    if level==1
      meanrow = meanrow*100;
    elseif level==2
      meanrow = findhigherbranch(branch)*100+meanrow*10;
    else
    meanrow = 100;
    end


    % Sets up the current list based on the branch
    currentpos=find(grpmarker==meanrow);
    currentlist=data(currentpos);
    ncurrent=length(currentlist);

    % Finds three means which are used to cluster values in the current branch
    lmmed=length(currentlist)/2;
    lmsma=lmmed-ceil(length(currentlist)/8);
    lmlar=lmmed+ceil(length(currentlist)/8);
    m(1,meanpos)=currentlist(lmsma);
    m(2,meanpos)=currentlist(lmmed);
    m(3,meanpos)=currentlist(lmlar);

    % initialize control variables (rm must not == oldm)
    oldm=0;
    rm=1;

    kdis=zeros(ncurrent,nummeans); %matrix for the distance from all values in
the
      %currentlist to each mean
```

```
    pass=0; %keeps track of how many passes before there is no change in any
mean



    while oldm~=rm;

      %Find the distance from each element in currentlist to the
      %means at meanpos.
      pass=pass+1;
        for i = 1:ncurrent,
            for j = 1:nummeans
                        kdis(i,j)=sqrt((currentlist(i)-m(j,meanpos))^2);
            end
      end

      [D posofnearest]=min(kdis'); %posofnearest is a vector with
                        %length(currentlist) and values of 1, 2, or 3
                        %D is irrelevant, as the values can be referenced
                        %using data


      % replaces the current 3 values at meanpos
      % if the means are the same then the loop will terminate
      oldm=rm;
      m(1,meanpos)=mean(currentlist(find(posofnearest==1)));
      m(2,meanpos)=mean(currentlist(find(posofnearest==2)));
      m(3,meanpos)=mean(currentlist(find(posofnearest==3)));
      rm=m;
  end  %WHILE
disp('passes:');
disp(pass);

  % saves the values found in posof nearest in vector grpmarker
  % grpmarker denotes position in entire tree rather than just the current
branch
  if level==0
      grpmarker=posofnearest.*100;
  elseif level==1
      grpmarker(currentpos)=grpmarker(currentpos) + posofnearest.*10;
  elseif level==2
      grpmarker(currentpos)=grpmarker(currentpos) + posofnearest;
  end

      meanpos=meanpos+1;
  end %branch

end %level

% The following procedure finds one mean for each of the remaining groups
% these means will be needed in the Fukunaga branch and bound tree search
mpos=13;
      for grp1=100:100:300
            for grp2=10:10:30
                  for grp3=1:3
                        g=grp1+grp2+grp3;
                        mpos=mpos+1;
                        m(2,mpos)=mean(data(find(grpmarker==g)));
                  end
            end
      end
```

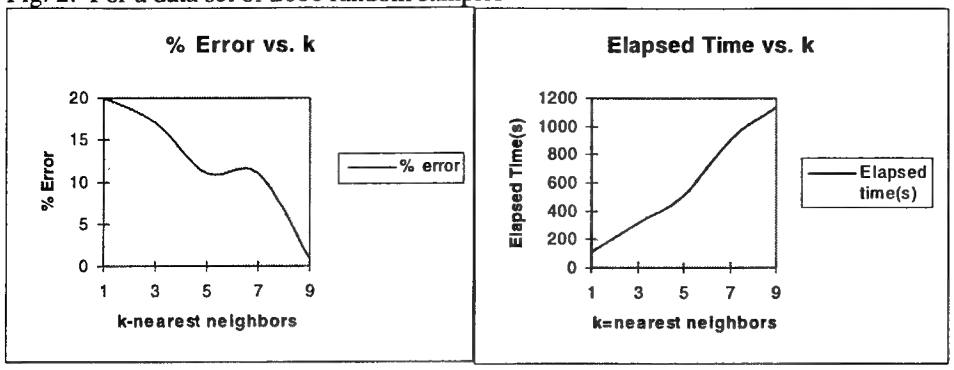
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% find radius of each cluster %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
r=zeros(1,40);
dobsm=r;
meanpos=1;
for grp=100:100:300
      meanpos=meanpos+1;
      grpmk00=(fix(grpmarker./100))*100;
      current=data(find(grpmk00==grp));
      dists=sqrt(current-m(2,meanpos)).^2;
      r(meanpos)=max(dists);
end
for grp1=100:100:300
      for grp2=10:10:30,
            meanpos=meanpos+1;
            grpmk0=fix(grpmarker./10)*10;
            grp=grp1+grp2;
            current=data(find(grpmk0==grp));
            dists=sqrt(current-m(2,meanpos)).^2;
            r(meanpos)=max(dists);
      end
end
for grp1=1:3
      for grp2=1:1:3
            for grp3=1:1:3
                  meanpos=meanpos+1;
                  g=grp1*100+grp2*10+grp3;
                  current=data(find(grpmarker==g));
                  dists=sqrt((current-m(2,meanpos)).^2);
                  r(meanpos)=max(dists);
            end
      end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

elapsedtime=cputime-timestart;
disp('elapsed time');
disp(elapsedtime);
```

The clustered data can now be searched by using Fukunaga's Branch and Bound search method{2}.

Results

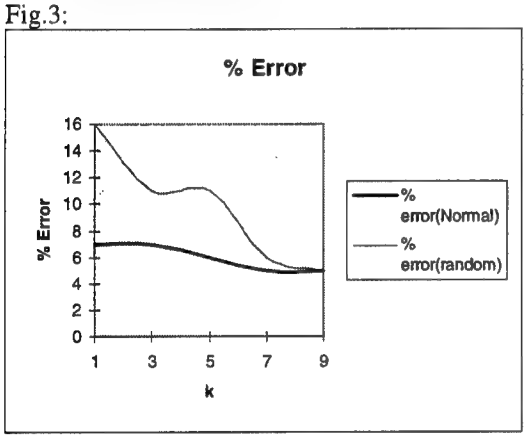
The simple k-nearest neighbor search is time consuming and inaccurate however, by increasing k and using the voting method, accuracy can be increased but only at a substantial cost of time. The following graphs show results of a test using 2000 random points. This test classified positive or negative values, not voice classes. It is, however, a good example of the inefficiency of the "Brute Force" method.

Fig. 2: For a data set of 2000 random samples



The observation set was then modified from completely random to a set of two normal distributions. This made checking the data much more efficient, eliminating the apparent error caused by using values to classify the data. The change is shown in the graph below.

Fig.3:



(Elapsed time showed an insignificant change.)

Elapsed time for multi-dimensional data is multiplied by the number of dimensions, results shown in figure 2 include any gains in elapsed time that would result from using the Linear Discriminant program. After the data has been projected to one dimension, we can apply the clustering algorithm. The k-means clustering method is quite efficient. For the 1 by 4000 voice data, the process took an average of 51 seconds. The clustering program should give all the data necessary for the Fukunaga Branch and Bound algorithm. The Fukunaga program is incomplete and untested, so it does not appear in this paper. However, preliminary results showed a great increase in efficiency. The unfinished program only applied rule 1 of Fukunaga's Branch and Bound algorithm and found only the nearest node, not the actual nearest neighbor. The program also did not classify the results so accuracy was not calculated.

Conclusion

The k-nearest neighbor classification method is not, perhaps, the best technique available for use in pattern classification. Accuracy can only be attained by a sacrificing large amounts of time. However, it is fairly simple and it is possible to maximize efficiency by applying Fisher's Linear Discriminant and using a clustering algorithm. Without using clustering accuracy must be sacrificed if speed is desired.

References

{1} Richard O. Duda & Peter E. Hart, *Pattern Classification and Scene Analysis*. New York: John Wily and sons, 1973 pp 114-118.

{2} Keinosuke Fukunaga and Patrenahalli M. Narendra, "A Branch and Bound Algorithm for Computing k-Nearest Neighbors." *IEEE Transactions on Computers*, pp 750-753, July 1975.

THE MULTI-TEMPORAL TRAINABLE DELAY(MTTD) NEURAL NETWORK ARCHITECTURE

Patrick X. Fitzgerald

Holland Patent High School
9601 Main Street
Holland Patent, NY 13354

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

THE MULTI-TEMPORAL TRAINABLE DELAY(MTTD) NEURAL NETWORK ARCHITECTURE

Patrick X. Fitzgerald
Holland Patent High School

Abstract

This paper presents an analysis of the Multi-Temporal Trainable Delay Neural Network architecture. The analysis was accomplished using multiple tests of an MTTD neural network simulator with a dual layer neuron scheme. Each test was done on multiple different dual-layer networks. From the test results the capabilities of the neural network was determined. The analysis tested the abilities of the network to learn and recognize temporal patterns. During the analysis of the architecture several changes were made to increase the capabilities and accuracy of the architecture. The network was run on a UNIX Sun SPARC 20 workstation running operating system Solaris 5.5. The program code was written in ANSI standard C and compiled under the same system software.

THE MULTI-TEMPORAL TRAINABLE DELAY (MTTD) NEURAL NETWORK ARCHITECTURE

Patrick X. Fitzgerald

Introduction

In the past few years, neural networks have become a powerful computational tool. A neural network is a computational model composed of a series of neurons and the connections between these neurons called synapses. A powerful application of neural networks is pattern recognition. A basic pattern recognition neural network is made up of two layers of neurons and the synapses connecting them. The first layer of neurons is the input layer. The input layer consists of input neurons. Input neurons are how external data is entered into a network. External data is translated by the input neuron into the network as a signal. The signal travels along synapses connecting the input neuron to each of the output neurons. Along this synapse a modification is done to that signal. The modified signal arrives at the output neuron. Each output neuron has a threshold. The threshold is the required signal strength for that output neuron to fire. When the threshold is exceeded a signal is transmitted by that output neuron. The strength of the signal transmitted by the output neuron, in pattern recognition neural networks, is the strength of the recognized pattern. Pattern recognition neural networks use these output firings to alter the modifications along each synapse. Additional layers of neurons can be placed between the input and output neuron. These internal layer are called hidden layers, because their signals do not reach the user directly.

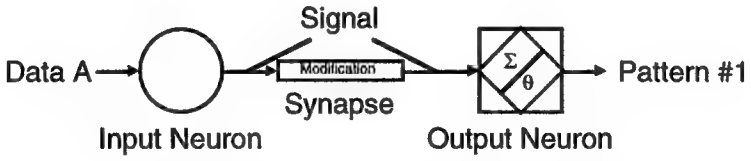


Figure 1: Neural Network Input, Output Pair

Most traditional neural networks use a weighting scheme. The strength of the signal is multiplied by the weighting value of that synapse. If the weighting value is greater than one, then the strength of the signal will increase. If the weighting value is less than one, then the strength of the signal will decrease. If the weighting value is one, then the signal strength will remain unchanged. Each modified signal arrives at another neuron. The signal arrives with other signals from other synapses, the strengths of all the incoming signals are summed. If the sum of the strengths exceeds the threshold the neuron will fire. The neuron will fire and its signal will travel along its synapses to other neurons. If the firing neuron is an output neuron the signal is output to the user.

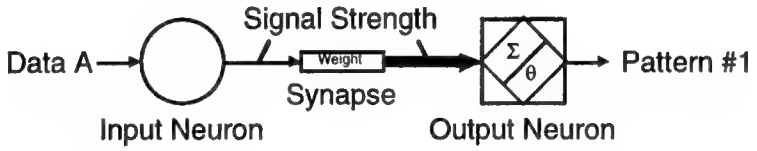


Figure 2: Traditional Weighting Scheme Input, Output Pair

The MTTD(Multi-Temporal Trainable Delay) neural network architecture uses a scheme of signal pulses and time-delays to specifically learn and recognize temporal event patterns. Each input neuron recognizes a different event, when the event is recognized by the input neuron the neuron fires. The strength of the signal fired changes over time, the signal strength is a pulse, when the event occurred is the peak of the pulse when the In the MTTD the modification along each synapse is not the traditional weighting of a signal's strength. Instead the signal is delayed for a simulated length of time. The signal travels as a signal pulse. The signal does not remain in the next neuron as it does in the traditional weighting scheme, the signal travels through the neuron, although not to other neurons. The threshold of the neuron will be exceeded when the signals arrive at the same time. The difference in time delays between neurons will be equal to the temporal difference between events in a learned pattern. The first event will have the longest time delay and the last event will have the shortest time delay.

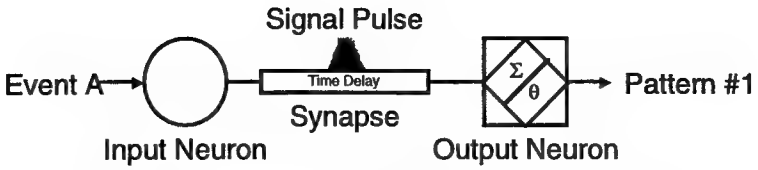


Figure 3: MTTD neural network Input, Output Pair

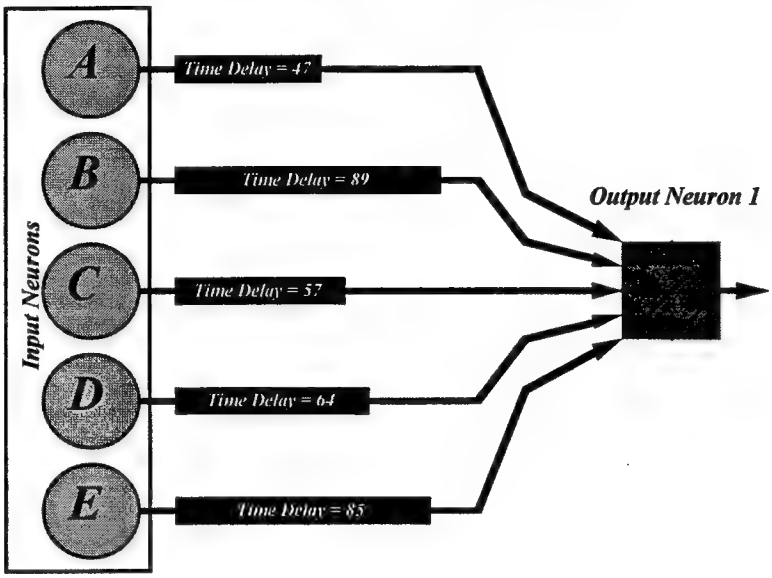


Figure 4: Trained MTTD 5x1 neural network

In Figure 4, this "5x1"(Five input neurons and one output neuron) network has been trained to recognize the following pattern. The output neuron will fire 47 seconds after event A occurs. The pattern could be trained with the example data and event. The network would then recognize the pattern B-4-E-21-D-7-C-10-A(Event - Time Difference - ...)

Event	Relative Time Difference		Example
Event B	Beginning	0	12:45:00 PM
Event E	+4 seconds	+4	12:45:04 PM
Event D	+21 seconds	+25	12:45:25 PM
Event C	+7 seconds	+32	12:45:32 PM
Event A	+10 seconds	+42	12:45:42 PM

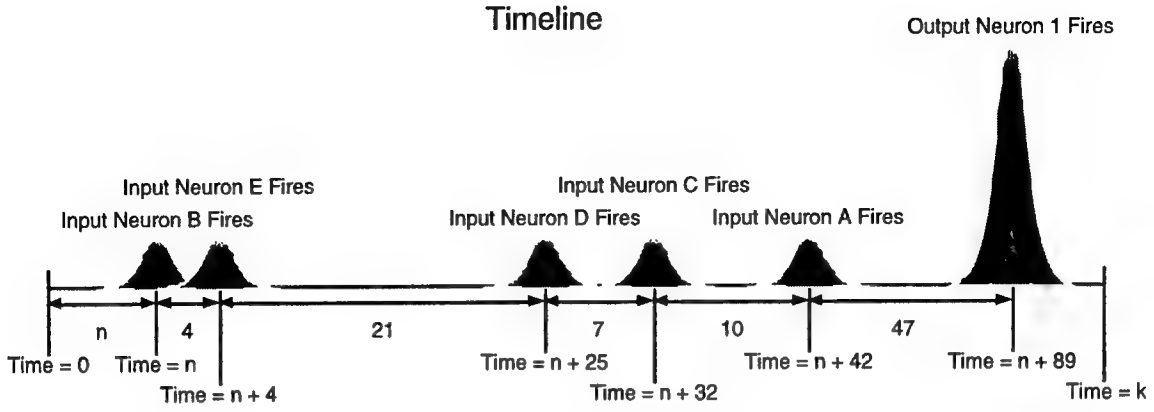


Figure 5: Timeline of Events

Discussion of the Problem

The objective of my summer project was to analyze the Multi-Temporal Trainable Delay neural network architecture. The use of an MTTD neural network architecture is in learning and recognizing temporal event patterns in time-stamped data. Temporal pattern recognition has applications in many areas, any were from analyzing combat tactics, to recognizing systematic "hacking" attempts on a computer system. The MTTD neural network was a prototype program established by Rome Laboratory. The program was established in theory, and had then been written in ANSI standard C. The prior testing of the architecture had been relatively simple, and had not been specifically oriented at testing the program capabilities and limits. To analyze the program tests were specially designed to test some of the architectures learning and recognition capabilities and limits. Due to the nature of any prototype program, multiple "bugs" had to be removed, and minor implementation problems corrected.

Testing

The first area of testing focus was on the networks ability to learn. This testing focused on what kind of patterns the network could be trained to recognize, and how well it would recognize those patterns. The training tests were done with patterns that had a reduced number of events in them and training that used multiple similar training patterns. Patterns that had a reduced number of events are patterns that have only some of the events recognized by all the input neurons. Using multiple similar patterns for training to have a single output neuron learn a generalized pattern, is beneficial when the true pattern is not known. Using multiple similar patterns to train multiple output neurons allows the network to distinguish minor differences between similar patterns. A desirable network is not required to use all the input neurons, and can both generalize and distinguish between similar patterns, depending on the circumstances.

Once a pattern had been learned, the second area of testing focus was the networks ability to recognize variations and altercations on the patterns it had learned. Tests on the networks ability to recognize generalized patterns and the effects of random noise on the output stimulus were the network recognition tests. The tests on generalized pattern recognition were done with test patterns that were missing events, and patterns that had their event times altered. The tests on the effects of noise, used random events within a range of random times. The random events were events recognized by the neural network's input neurons, the time range was ten percent before the pattern started and ten percent after the pattern ended.

Capability to Learn: Reduced Training Patterns

The first testing consisted of training the network for patterns that did not contain all the events that were recognized by the input neurons of the network. Training an output neuron to learn a reduced pattern does several things. The maximum stimulus that the output neuron sees during training is less than the maximum possible stimulus on a full event pattern, a pattern that uses all the events recognized by the input neurons. This causes the threshold of the output neuron to be less than the full event pattern's threshold. The time delays along the synapses from the input event neurons that are not in the pattern to the output neuron learning the reduced pattern, are not changed during training.

Results from recognition tests showed that the output neuron learning the reduced pattern is not exclusively trained for the desired event pattern. The undesired events of that pattern will influence the strength of the output stimuli. The time delays along the synapses to undesired events remains unchanged from what the time delay was at its initial setting. The undesired event, when occurring at a specific time, will influence the strength of the output stimuli. When the undesired event occurs at the right time it is as influential to the output neuron as a desired event. Since the recognition threshold of the output neuron will be lower on a reduced pattern, the undesired event patterns can cause a greater output stimulus than the desired event pattern. The greater the number of undesired input events the higher the number of undesired patterns that will cause an output stimulus.

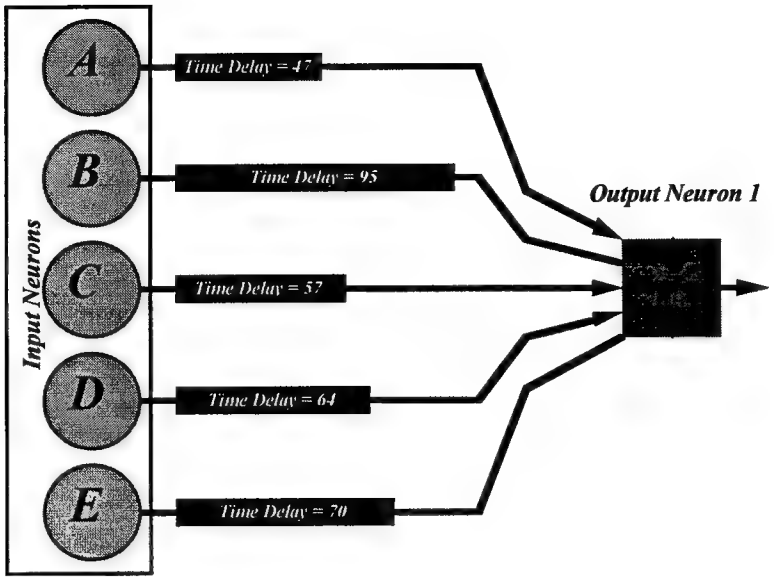


Figure 6: Trained 5x1 MTTD neural network for pattern D-7-C-10-A

The trained network above will also recognize with the same strength patterns E-6-D-17-A, B-15-E-6-D, etc. The output neuron will have a higher stimulus with 4 or 5 event patterns using the undesired events B and E.

Learning Capabilities: Multiple Similar Training Patterns

Multiple similar training patterns were used to test both the networks ability to learn a single generalized pattern, and the ability of the network to learn to distinguish patterns with extreme precision and accuracy. Although the two tests seem like the exact opposite, the neural network can be altered by modifying values to do both. The threshold ratio is one of the important values that can be adjusted to allow multiple training patterns to either be learned as one generalized pattern or multiple patterns. A high threshold ratio will allow the output neurons to train accurately, because the strength differences between the right pattern and the wrong patterns will be small, but a high ratio will be able to distinguish them. A low ratio will allow the network to settle easily. The learning rate value also effects the accuracy/generalization learning capabilities. A very low learning rate will help the patterns separate easily, a medium rate will allow the patterns to be learned in a generalized way, and a very high rate will not allow settling, the alteration continue forever.

Learning Capabilities: Training for a Single Event Pattern

Multiple similar training patterns were used to train a single output neuron. The network would learn the similar patterns to a point of resonance, where the differences in time between patterns would cause the time delays to lengthen and shorten. The network would only settle when there was no longer any modifications being done to the time delays. The network would never settle, even though the delays were changing only slightly. The code was changed so, if the modifications were "small enough" then the network would settle. The network will learn to recognize one pattern from multiple similar input training patterns. The threshold on the output neuron is less that the threshold of an output neuron that learned using only one training pattern. The learning rate effects the size of the modification resonance, therefore also effecting the required similarity of the patterns. Even though the resonance of the modifications is small, it never decreases. The modifications are always being changed toward another pattern. When the network settles the pattern learned depends most on what modification to the time-delays that were done last. If the learning rate is small this effect will be minimized.

Learning Capabilities: Training for Multiple Event Patterns

Multiple training patterns were used to train multiple output neurons similar patterns with precision and accuracy. Initially, the network would not settle, because the threshold ratio was too low. All the patterns could exceed the threshold value and fire each of the neurons. If the threshold ratio was increased enough the network would settle without learning all the patterns. Multiple output neurons would start to learn the same pattern. Each output neuron started "fighting" over that pattern. One of the neurons eventually would win and learn that pattern to completion. The other neurons would have learn that pattern only partially. The pattern, once learned strongly enough is essentially taken out of the training process. The winning neuron always wins that pattern. The other neurons had learned the previous pattern too well to start learning the other patterns. The other patterns were never learned. No modifications were done on those neurons, so the network settled without learning all the patterns. The code was modified to wait until the

network settled. If some of the patterns were not being learn the threshold on an untrained neuron was decreased. The untrained outputs were allowed to learn the unlearned patterns. The network can learn multiple similar patterns and distinguish them with accuracy.

The network will work only with an amount of trial and error. Multiple values had to be altered manually before the network would settle. In order for the network to learn independently each pattern takes a great deal longer than a network only learning from one pattern. The learning rate and the threshold ratio are independent of each other, but they do depend on the specific patterns that are being used to train. These values must be specifically set for each trial. The "small enough" value must be too small, so that multiple training is allowed.

Recognition Capabilities

The MTTD neural network was designed to learn and recognize multiple temporal patterns. The testing on the recognition capabilities of the network was done on "well learned" patterns. A "well learned" pattern means that the pattern uses all events that there are input neurons for, and that the output neuron has trained the time delays so that the signal strength from a testing of the training pattern was within 95% of the maximum possible stimulus that could be seen from that pattern. The recognition generalization capabilities of the network were tested using partial patterns and altered event times. The testing of partial patterns, used event data patterns that were missing events, this recognition testing corresponds similarly to the reduced pattern training testing in the kind of patterns used. The altered event times testing used test patterns that had events that occurred at similar times, but each pattern had events that were off a varied amount of time.

Recognition Capabilities: Generalization: Partial Patterns

Testing was done using multiple learned event patterns that had events missing from the pattern entirely. Using the example in figure 4, the test patterns for partial pattern recognition could have been B-4-E-21-D-7-C, E-21-D, B-32-C-10-A, etc. Each missing event lowered the output signal strength by half the original amplitude of the event pulse. The number of events that could be missing and still have enough output signal to fire the output neuron depended on the threshold ratio and the number of events in the pattern. In order for the output neuron to fire the a ratio of accurate events equal to or greater than the threshold ratio must be present. Partial patterns were recognized by the output neuron, but only if there was a percentage of events still present that was equal to or greater than threshold ratio.

Recognition Capabilities: Generalization: Altered Event Times

Testing was done on multiple learned event pattern that had a number of events with minor alterations of event times. The output neuron would still fire if the alterations were few and/or small enough. The alterations were independent of the event in "well learned" networks. This means that if event A was off a second, or event B was off a second the signal strength would decrease the same amount. If multiple events were altered the signal strength would decrease the sum of all the individual alteration amounts. Only the time differences between the desired events and the tested events affected the stimulus. The effective reduction of stimuli was dependent on the time difference and the width of the signal pulse. The width of the pulse below was five and if the event time was altered by five or more the output strength was the equivalent as when the event was not even there.

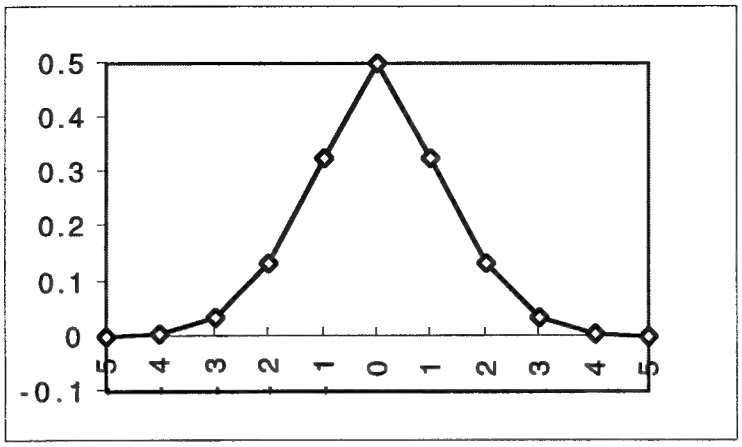


Figure 4: Effect of Alter Event Times

Recognition Capabilities: Effects of Noise

The recognition testing on the effects of noise on signal strength in the MTTD neural network was done by using random recognizable events and effective random times. The number of "noisy" events was varied. A recognizable event is one that the network has an input neuron for. Effective times are times within 10% of the range of the initial training pattern. The number of events did not directly effect the stimulus strength, although the greater the number of "noisy" event the greater the chance that they would be effective "noisy" events. An effective "noisy" event is an event that falls within the pulses range of its corresponding event. In other words if a "noisy" event A fell within five seconds of the event A that was part of the learned pattern then the signal strength was increased. The output stimulus's strength was increase by one half the amplitude of the event signal minus the amount that an altered event would have decreased the signal. Any of the correct event noise outside the correct event pulse width did not influence the pulse strength.

Conclusions

The MTTD neural network architecture was designed to learn and recognize temporal event patterns. In the testing of the learning capabilities, several items of interest appeared. First the network can not exclusively learn reduced event patterns. Second, the network relies too heavily on human input and alterations. The abilities of the network to learn is severely hampered, by the "need" of network to have the correct values supplied by the user. The program is not very user friendly, the number of variables the user must know to train the network is confusing. The number of patterns to be recognized needs to be known also. The recognition testing of the neural network showed that the generalization capabilities of the network are linear and a adjustable by changing the pulse width and the amplitude. The tests on the effects of noise, show that noise can effect the output stimulus. The effective noise is limited, but still present.

Some possible solutions or alterations to minimize this items could be made. By adding some form of a weighting device could eliminate the effects of undesired events in reduced patterns. The network could use dynamic variables that change over the course of training to eliminate the reliance on the user. The amount of generalization is determined by the desired method, the pulse shape could be changed. The network could adapt the pulse width using the relative time differences, and proportion its size accordingly. To reduce effective noise input neurons could have a maximum firing power. This would plane the pulse off, effectively keeping the amplitude of pulse at this maximum.

References

Barth, Scully, *A Functional MTTD Neural Network Simulator for Situation Assessment.*

Jumper, Eric J., *A Multi Temporal Trainable Delay Neural Network,* IEEE International Conference on Neural Networks, Orlando, FL, Vol. I, 1994.

O'Neil, Barth, Scully, *Introduction and Application of the Multi-Temporal Trainable Delay Neural Network.*

O'Neil, Paul A., *A Self Organizing Neural Based Representation Generator and the Multi-Temporal Trainable Delay(MTTD) Neural Network.*

Wasserman, Philip D., **Neural Computing: Theory and Practice**, Van Nostrand Reinhold, New York, 1989.

Zeidenberg, Matthew, **Neural Networks in Artificial Intelligence**, Ellis Horwood, England, 1990.

RF MODULE LIFE TEST SYSTEM DESIGN

Daniel Grabski

Holland Patent High School
Thompson Road
Holland Patent, NY 13354

Final Report for
High School Apprentice Program
Rome Laboratory
Rome, NY

Sponsored by
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory
Rome, NY

August 1997

RF MODULE LIFE TEST SYSTEM DESIGN

Daniel Grabski
Holland Patent High School

Abstract

This RF module life test system is a conversion from an earlier life test system. The system life tests RF modules under heat stress for long periods of time, and records data on the module's RF output. The new system is designed to be more versatile than the earlier version through improved RF switching and reduction of unnecessary components. The new life test system was also tested for attenuation through its RF input and output channels at frequencies in the C and X bands (2 to 12 GHz).

RF MODULE LIFE TEST SYSTEM DESIGN

Daniel Grabski

Introduction

The life test system is designed to test RF modules for failure over long periods of time under stress. The original system tested over a wide range of frequencies, however, many trays of RF equipment were needed to control and measure a module's RF properties. This new system is designed to eliminate unnecessary components to make the testing simpler and more reliable, while also incorporating test control through a graphical interface.

Description of Life Test System

The physical system consists of six rack-mount cabinets. The life test system has the potential to test up to ten modules under stress at the same time. The system provides each module with a dedicated power supply, heater, thermocouple, RF input and RF output. The system also can supply six additional modules with a power supply, heater, and thermocouple, which brings the number of modules under stress at any time to sixteen. The sixteen module locations are arranged in four groups, with each group of four in a sealed metal box. All four test boxes reside in the fourth cabinet.

The heaters heat each module under test, and the thermocouple regulates the module's temperature. The heater controllers reside in the third cabinet.

The power supply boards are custom made for this application. Using three commercially available power supplies, the power supply boards provide a ground, regulated input voltage, and regulated bias voltage for all modules under test. The boards incorporate potentiometers which are used to calibrate the output voltages from the boards. The boards are removable for easy replacement. All power supply boards reside in the second cabinet, with the commercially available power supplies below the heater controllers in the third cabinet.

An IBM-compatible computer with an HP-IB interface card is in the first cabinet. The computer runs HP VEE, a program from Hewlett-Packard designed to control test instruments over the Hewlett-Packard Interface Bus (HP-IB). Each test instrument in the life test system and the computer is connected with HP-IB cables, so the HP VEE program can send signals through the interface card and over the HP-IB cables to control all the instruments used in the life test system. HP VEE incorporates a graphical user interface and programming features to simplify control of the system. The entire test system can be controlled through an HP VEE program.

The fifth and sixth cabinets hold RF generation, switching, and measuring equipment. The fifth bay holds four HP 11713A Attenuator/Switch Drivers and several trays of RF switching equipment. The equipment takes an RF input, splits it into ten input channels, switches these channels on or off, and then provides an input to the module under test. The input signal is generated from the HP 8341B Synthesized Sweeper. The input is split into the ten channels, and then goes to the module under test. From the modules under test, the signals travels through several couplers, each of which attenuates the signal, and then all ten outputs are joined together. The attenuators are needed to control a potentially large amount of power transmitted from amplifiers, if they are being tested.

Data

Using the the HP 8341B Synthesized Sweeper, I applied a 14dBm (15dBm at the sweeper, minus 1.0 dBm attenuation from an isolator placed on the sweeper's output) signal at various frequencies to the RF switching input and measured the output at the DUT box using the HP 8566B Spectrum Analyzer with reference level set at 14 dBm. Using these power figures, I was able to determine the attenuation of each input channel, from the signal input to the DUT. Using the same method, I applied a 14 dBm signal to the output of the DUT box, and measured the power at the power meter connection. I repeated all measurements at frequencies from 2 to 12 GHz. These measurements were performed on five of the system's ten RF channels.

This data can be valuable information for calibration of power readings. Since the attenuation of the switching circuit varies with frequency, this data must be used to produce accurate readings.

For the test data and graphs of signal loss for each channel, see pages 17-7 through 17-11.

Future Improvements

The major future improvement for the system is a redesign of the RF switch driving circuits. Currently, the life test system uses four HP 11713A Attenuator/Switch Drivers to drive the RF switches in the system. Also, some sets of switches do not switch independently. Through relays, all ten sets of RF switches can be switched independently using one HP 11713A Attenuator/Switch Driver [see Fig. 1 for circuit]. The three RF switches in any one set must switch simultaneously, and each set must switch independently. This circuit takes the output from the attenuator cable of a Hewlett-Packard 11713A Attenuator/Switch Driver and drives four HP 33312B RF switches and eight 8761B RF switches. Normally, the 11713A can control only two four-step attenuators and two RF switches, but by cutting the end connector off of both attenuator cables, the driver can be wired to drive the ten sets of RF switches.

Each 11713A driver has connections for two attenuator cables. Each attenuator cable has twelve wires: one wire is a +24Vdc supply, one wire is a ground, two wires are for one of the two switch drivers, and the other four pairs of wires are usually for the attenuator. In each pair of attenuator wires, one wire will be grounded, and one wire will be floating at a high resistance. In the pair of switch wires, one wire will be grounded, and one wire will be +24Vdc. The front panel buttons switch the state of the to wires in each pair: in the case of an attenuator wire pair, the wire in the corresponding pair that was grounded will then be floating at a high resistance, and the wire that was floating at a high resistance will then be grounded. Four buttons for one attenuator and one switch button controls all the pairs in one attenuator cable. See Figure 1 on page 17-12 for the circuit schematic.

Using one pair of attenuator wires, this circuit can control all three RF switches in one set. To drive the 33312B, the +24Vdc wire is connected to the C pin on the switch, one attenuator wire is connected to the 1 pin, and the other attenuator wire is connected to the 2 pin. So, there is always +24Vdc on the switch supply pin, and the ground alternates between pin 1 and 2, depending on the state of the front panel button. The button will then drive the switch.

To drive the 8761B switch, +24Vdc is applied to one input pin, and a ground is applied to the other pin. When the two inputs are switched, the RF switch is switched. The two 8761B switches in any set can be driven by one attenuator wire, two normally closed SPST relays, and two normally open SPST relays.

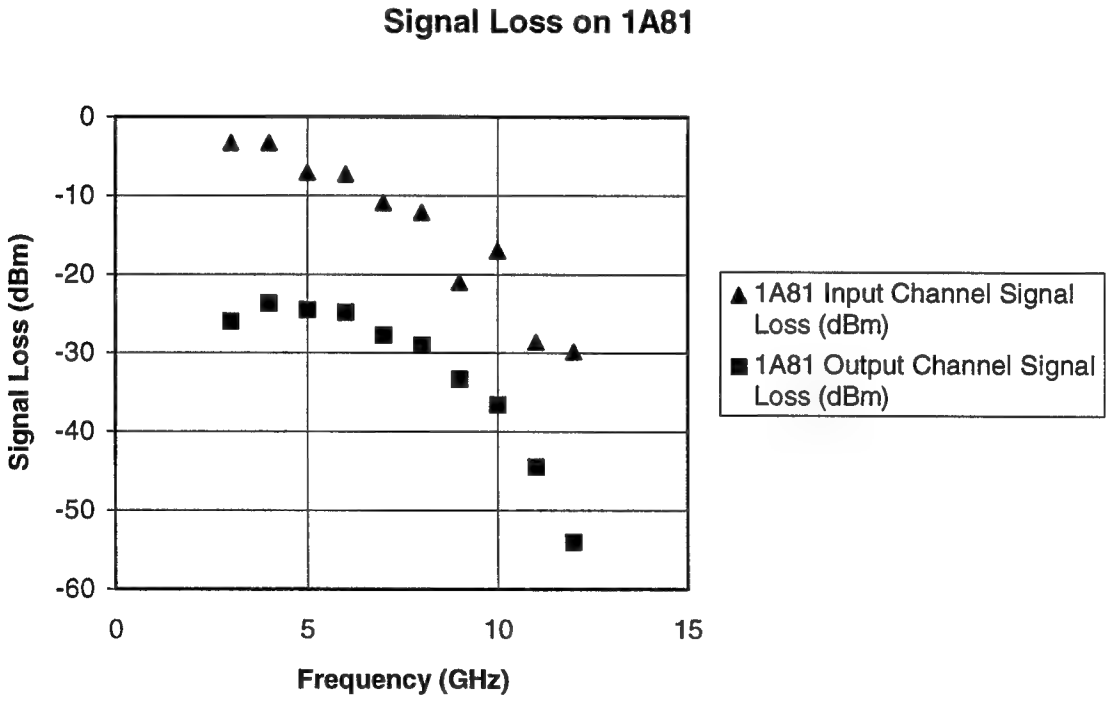
An independent +12Vdc power supply is used for the relays. When the attenuator wire is floating at a high resistance, the relays stay in their normal positions. When the attenuator wire is grounded, the relays switch, and the +24Vdc and ground inputs are flipped, which drives the RF switch.

Another improvement is a change of the power supply boards. Currently, the power supply boards are custom made for this system. Removing these power supply boards and using commercialy available power supplies would make the system less complicated, as it would also replace the commercial power suplies used to supply the power supply boards. Also, commercially available power supplies would eliminate the possibility of running out of the current custom power supply boards.

Test Data

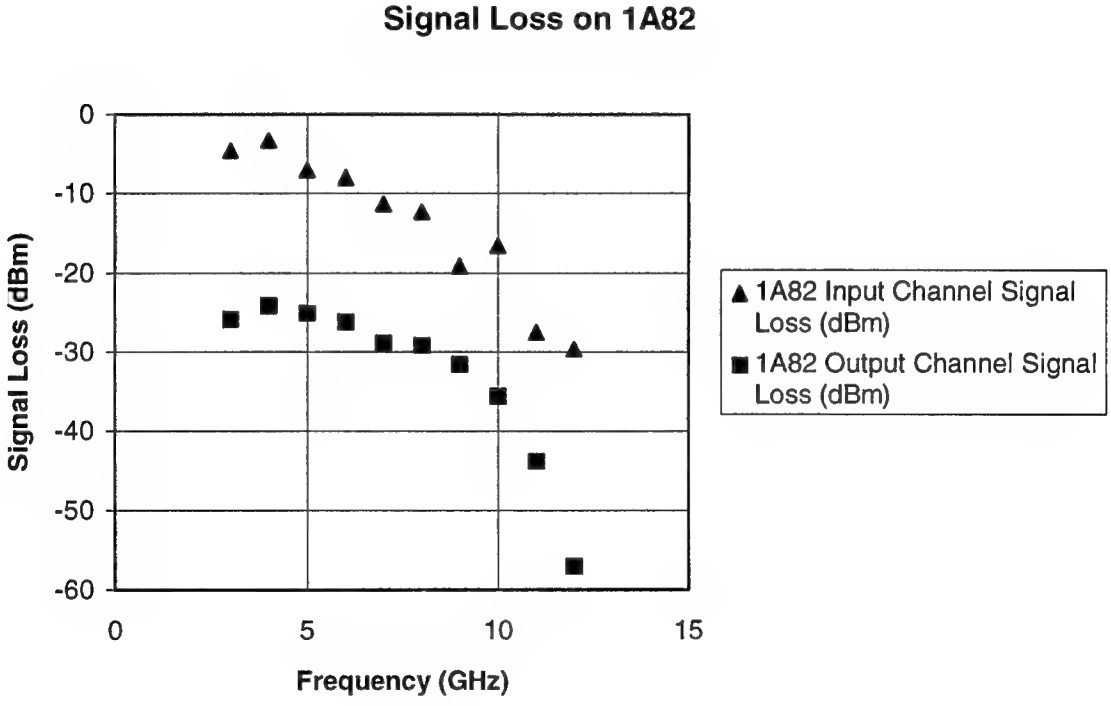
Channel 1 (1A81)

1A81 Frequency (GHz)	1A81 Input Channel Signal Loss (dBm)	1A81 Output Channel Signal Loss (dBm)
3	-3.3	-26.0
4	-3.3	-23.7
5	-7.1	-24.5
6	-7.3	-24.8
7	-10.9	-27.7
8	-12.1	-29.0
9	-21.0	-33.3
10	-17.0	-36.6
11	-28.6	-44.5
12	-29.8	-54



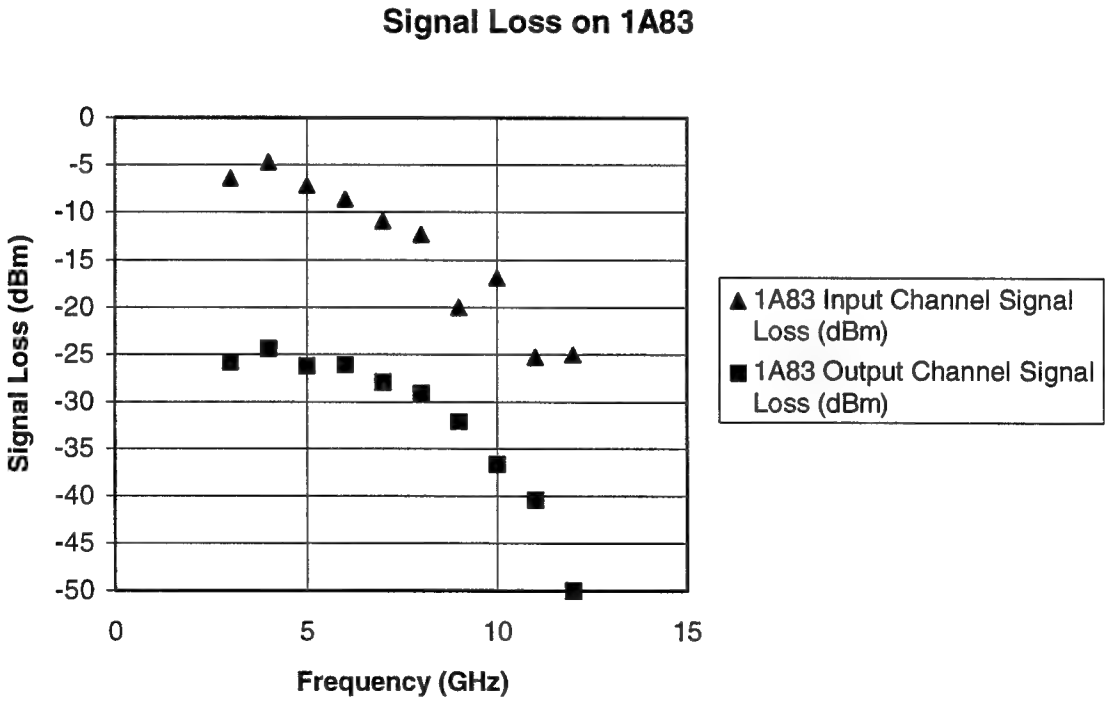
Channel 2 (1A82)

1A82 Frequency (GHz)	1A82 Input Channel Signal Loss (dBm)	1A82 Output Channel Signal Loss (dBm)
3	-4.6	-25.8
4	-3.3	-24.1
5	-7.1	-25.0
6	-8.0	-26.1
7	-11.3	-28.8
8	-12.3	-29.1
9	-19.1	-31.5
10	-16.5	-35.6
11	-27.4	-43.8
12	-29.6	-57



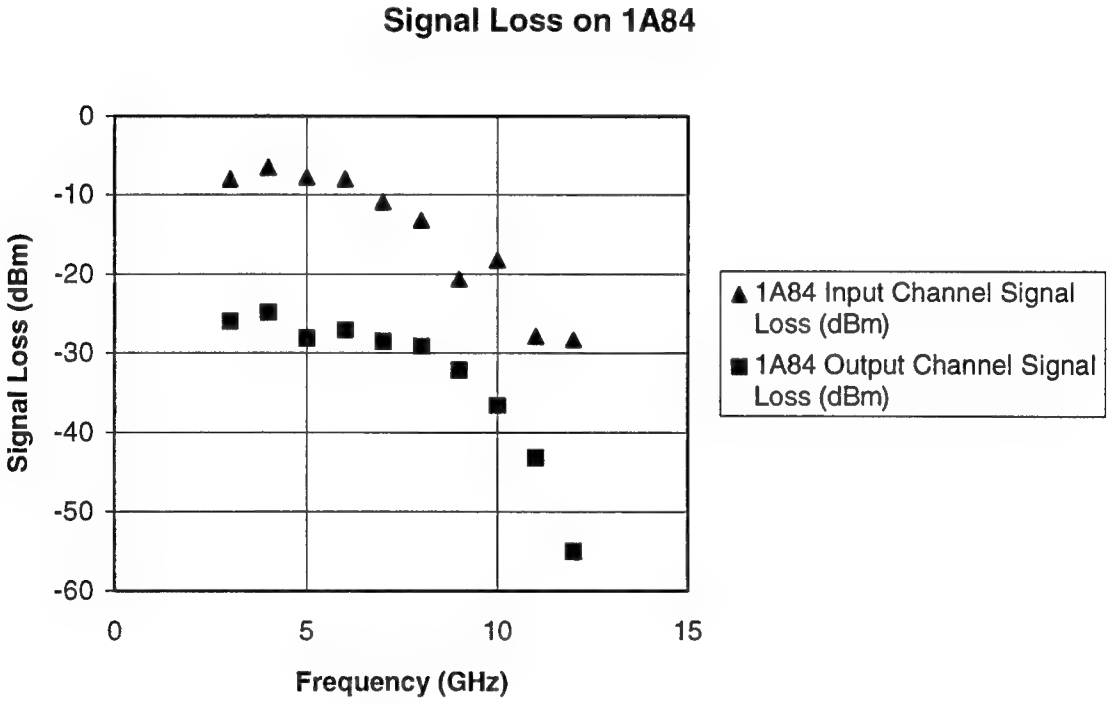
Channel 3 (1A83)

1A83 Frequency (GHz)	1A83 Input Channel Signal Loss (dBm)	1A83 Output Channel Signal Loss (dBm)
3	-6.4	-25.8
4	-4.7	-24.4
5	-7.2	-26.2
6	-8.6	-26.1
7	-10.9	-27.9
8	-12.3	-29.0
9	-20.0	-32.1
10	-16.9	-36.6
11	-25.2	-40.4
12	-25.0	-50



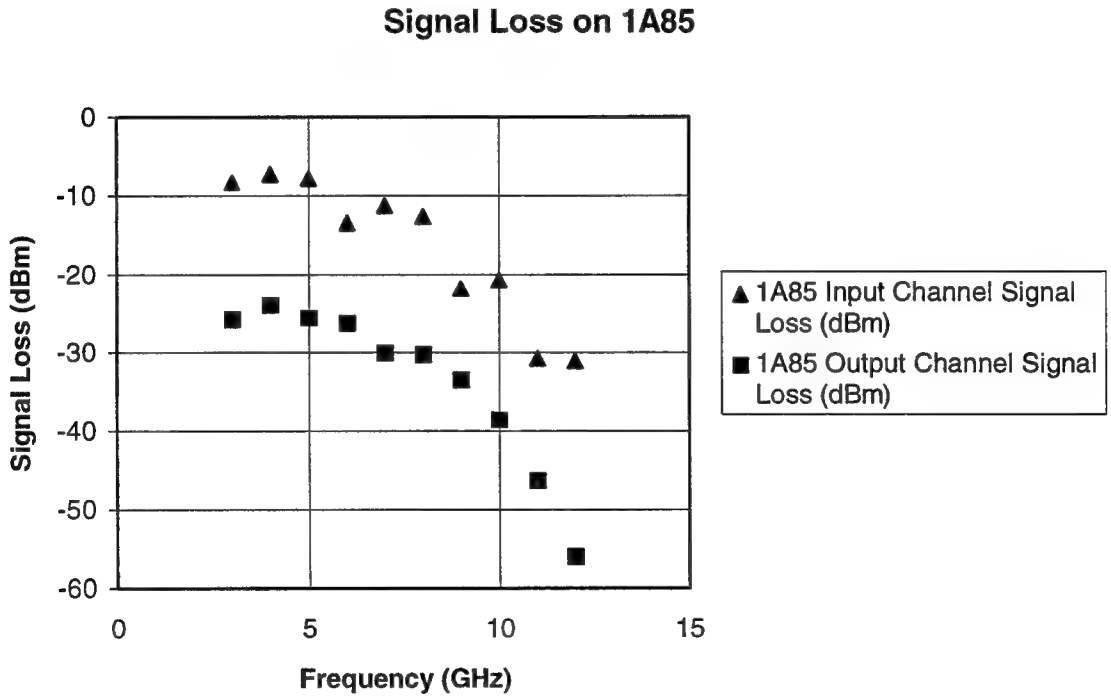
Channel 4 (1A84)

1A84 Frequency (GHz)	1A84 Input Channel Signal Loss (dBm)	1A84 Output Channel Signal Loss (dBm)
3	-8.0	-26.0
4	-6.5	-24.8
5	-7.8	-28.1
6	-8.0	-27.1
7	-10.9	-28.5
8	-13.2	-29.1
9	-20.6	-32.1
10	-18.2	-36.5
11	-27.8	-43.1
12	-28.3	-55



Channel 5 (1A85)

1A85 Frequency (GHz)	1A85 Input Channel Signal Loss (dBm)	1A85 Output Channel Signal Loss (dBm)
3	-8.3	-25.7
4	-7.3	-23.9
5	-7.8	-25.5
6	-13.4	-26.2
7	-11.2	-30.0
8	-12.6	-30.2
9	-21.8	-33.5
10	-20.7	-38.6
11	-30.7	-46.3
12	-31.0	-56



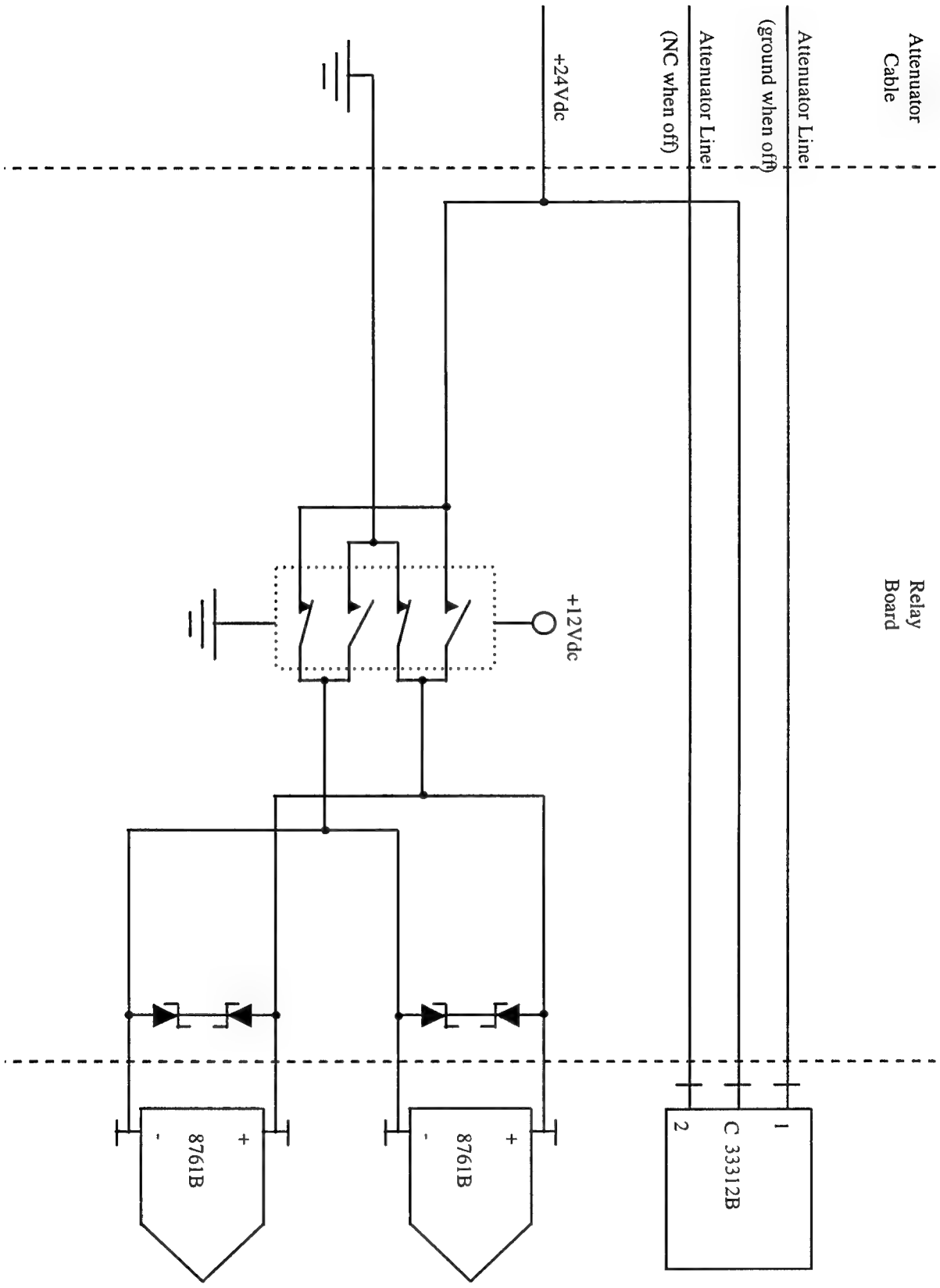


Figure 1. RF Switch Driver Circuit Schematic.

ANTENNA PATTERN MEASUREMENTS USING INFRARED IMAGING TECHNIQUES

Sandra L. Jablonka

Oneida High School
560 Seneca Street
Oneida, NY 13421

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

ANTENNA PATTERN MEASUREMENTS USING INFRARED IMAGING TECHNIQUES

Sandra L. Jablonka
Oneida High School

Abstract

Antenna pattern measurements using infrared imaging techniques were studied. An infrared test setup was used in an anechoic chamber to gather the measurement data. To collect different sets of data, minor variations were made to the setup. The data was processed by Matlab visual software in order to make comparisons concerning the modifications of the test setup. As a result of the processing and analysis, the configuration for antenna measurements was optimized.

ANTENNA PATTERN MEASUREMENTS USING INFRARED IMAGING TECHNIQUES

Sandra L. Jablonka

Introduction

Infrared (IR) techniques deal with the rise in temperature, when placed in an Electromagnetic (EM) field, of a lossy material. An IR camera reads the temperature data from the material and transforms it into the intensity of the field. The degree that the temperature of the material rises is related to the strength of the field -- the higher the temperature, the stronger the field. Microwave holography techniques determine complete field data from thermographic measurements. A hologram is an interference pattern with which it is possible to create the complete complex image. Two antennas are used to construct the pattern. One is the reference antenna and the other is the antenna under test (AUT) [1]. By making comparisons of the data gathered from different holographic test setups, the setup which optimizes the configuration for antenna measurements can be determined.

Setup

In the Rome Laboratory setup, a 36 element patch array antenna was used as the AUT. A 4 Ghz Standard Gain Horn was used as the reference antenna for the measurements and was placed at a 60° angle to the normal of the thermal paper. The thermal paper is a lossy material made of carbon-loaded Kapton. Two different types of this paper were used, one with a resistivity of 340 ohms/square and one of 1500 ohms/square. The paper is mounted on a 60 x 60 piece of art board with a wooden frame surrounding it for stability. About three meters below the thermal paper is the AGEMA 900 thermal camera. The IR camera system converts the radiance of the paper to surface temperature.

It averaged 32 frames of data to produce a stable image on the screen. The camera is cooled by liquid nitrogen which makes it impossible to turn it vertical 90°. This angle was needed in order to view the paper hanging above. Because the camera could not be turned vertical, a gold coated first-surface mirror was used that allowed the camera to remain horizontal. The mirror redirected the view of the camera. The mirror was obtained from Edmund Scientific was used [1]. Holding the reference antenna, AUT and thermal paper is fiberglass scaffolding. Because metal cannot be used with the setup, nylon bolts were used to fasten the scaffolding.

Methodology

To determine complete field data, holographic techniques can be used. A hologram is a pattern created by the constructive and destructive interference of two signals. Local maximas are constructive, whereas local minimas are destructive. It contains phase information. After a 180° phase shift of the reference antenna, the points that were constructive become destructive, and the points that were destructive become constructive. The reference antenna and the AUT are set up to radiate the paper that the thermal image is recorded from. A single source is split to feed the two antennas so that an interference pattern is generated on the thermal paper. The relative difference in phase of the fields at each point produces the interference pattern [1].

Numerous sets of data were collected. They were gathered at reference antenna angles of 45° and 29° relative to the plane of propagation of the AUT. There were also phase shifts of 0°, -9° and -20°. The data was plotted using a Matlab program written by Dr. John Norgard. The graphs show different pictures of the data including the intensity of the antenna near field, the intensity of the reference antenna, the intensities of two holograms and the amplitude and phase of the reconstructed near fields as well as others. However, the plots were in a generic form and needed to be modified for each

individual set of data. This included trimming excess data that was not needed in analysis and removing messy noise from the phase plots. Also the plots were rotated 90° to view the data on the opposite side of the graph. After the data was cleaned up and the graphs were printed, the data sets were analyzed for the generation of an optimal setup.

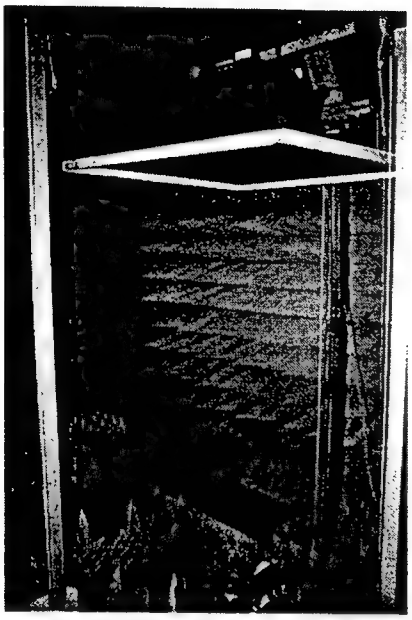


Figure 1 - IR Measurement Setup

Results

Of the various data sets processed, the first collected showed optimal results. The antenna near field data and the reference field plots were very similar throughout most of the data sets. Figure 2 represents the magnitude of the antenna near field of the AUT field only. Figure 3 displays the magnitude of the intensity of the reference field of the reference antenna only. Figure 4 shows a holographic plot which is the interference pattern of the near field of the AUT and the reference field radiating at the same time. The shapes of the plots were fairly regular and, when rotated, appeared to be symmetrical unlike later sets of data. Figures 5 and 6 represent reconstructed phase plots which have the least amount of cluttering noise surrounding them. These ideal measurements were

taken with the reference antenna at an angle of 45° and a 0° phase shift. As the phase shifted on the other sets, the amount of power decreased the magnitude of the antenna near field and the magnitude of both holograms.

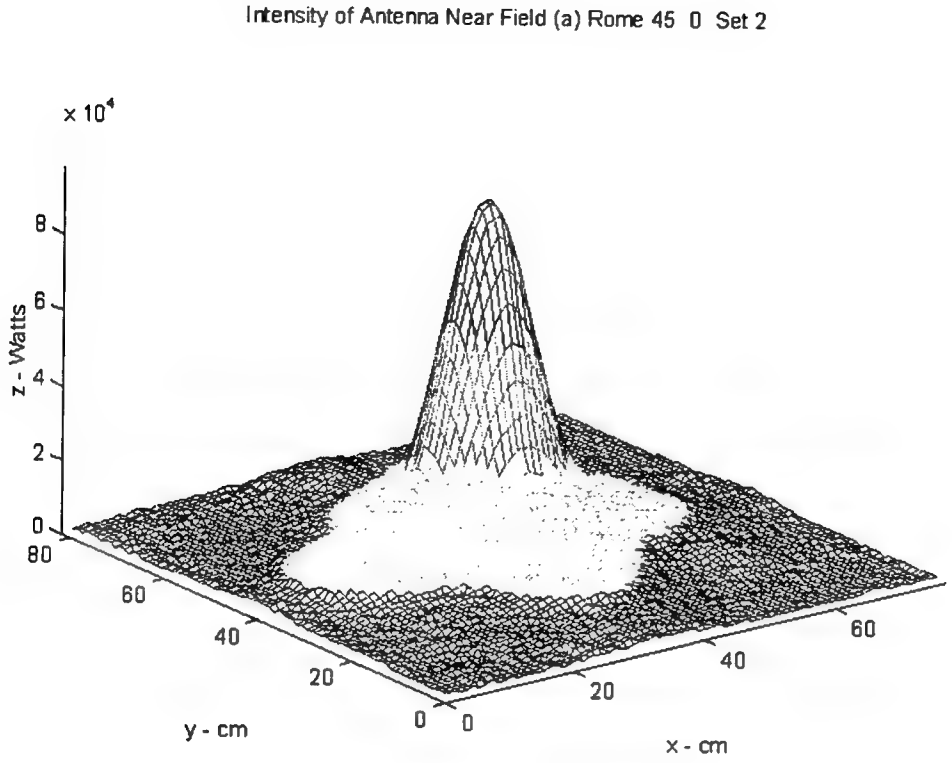


Figure 2

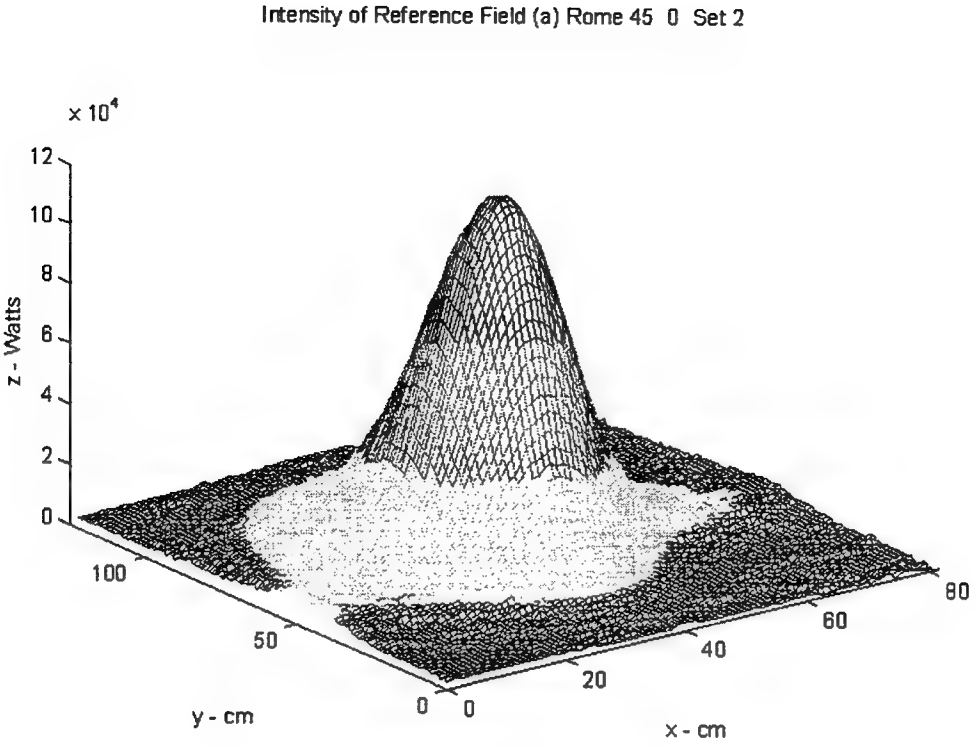


Figure 3

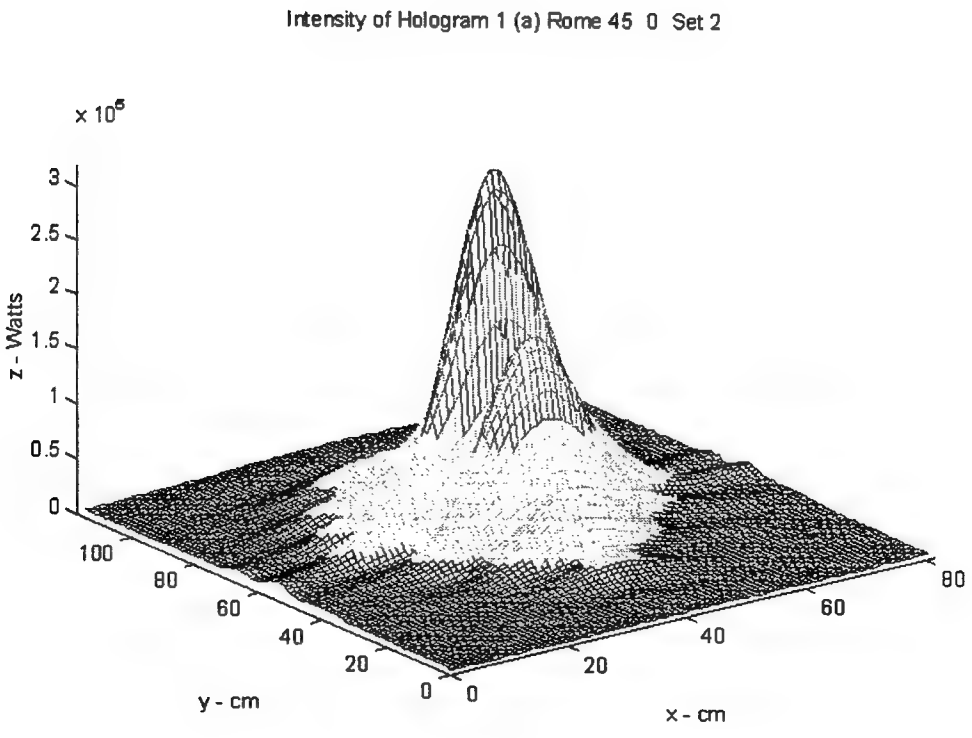


Figure 4

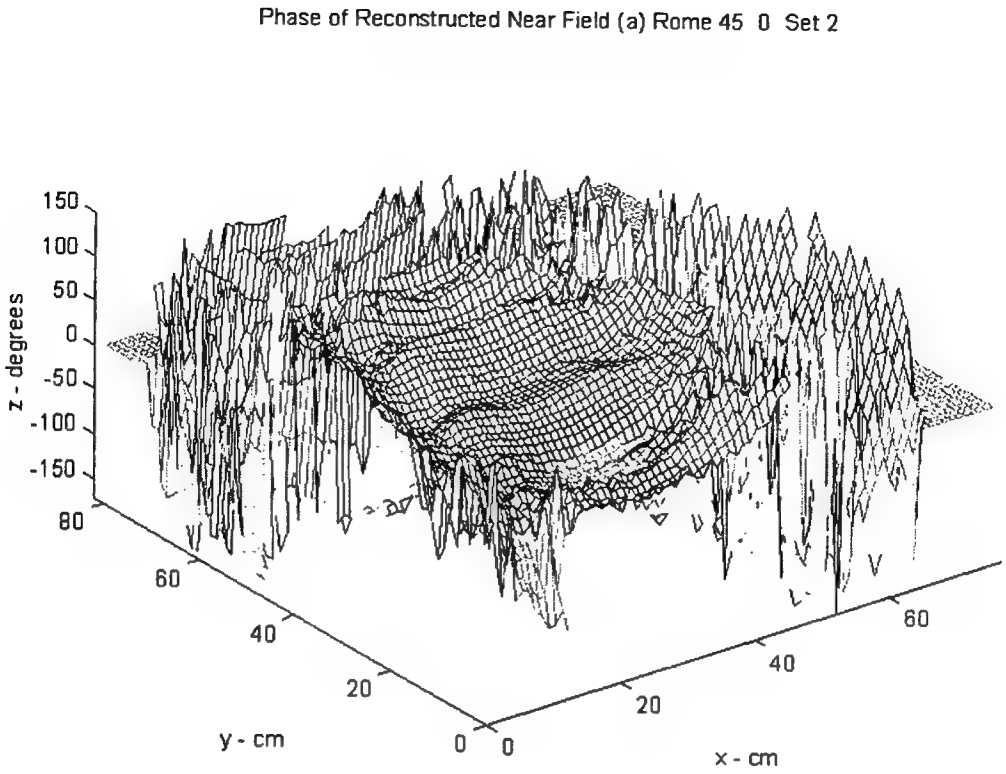


Figure 5

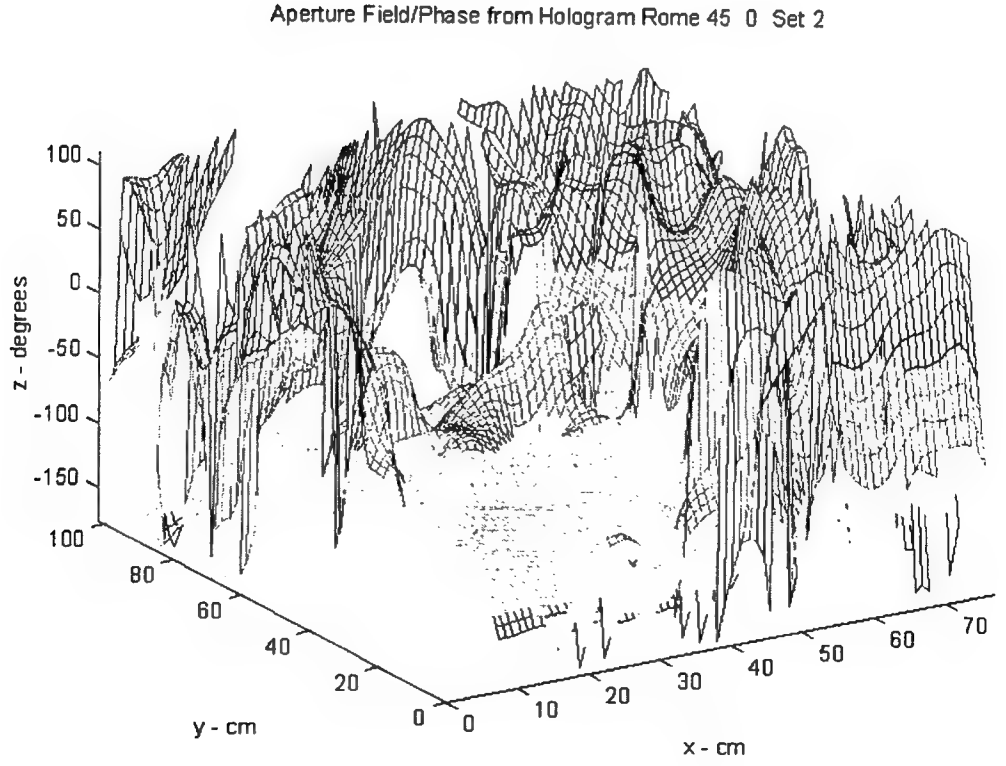


Figure 6

Conclusion

The IR camera setup is a method of making antenna pattern measurements. After being processed by Matlab the information, in graphical form, was easily compared to create the ideal measurement setup. The correct configuration enhances the measurements to be as useful as they can be.

Reference

[1] J.E. Will and J.D. Norgard, "Innovative C3I Technologies - Complex Magnitude and Phase Measurements of EM Fields from Transmitting Antennas Using Innovative Infrared Imaging Techniques", pp. 5-9, 43-47, Oct. 8, 1996.

A STUDY OF GENETIC ALGORITHMS

Colin M. Kinsella

Oneida High School
560 Seneca Street
Oneida, NY 13421

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

A STUDY OF GENETIC ALGORITHMS

Colin M. Kinsella
Oneida High School

Abstract

I worked on making a simple genetic algorithm with programming in Microsoft's Visual C++ version 4.0. I tried to write the program inside a console window and as a windows application. Each presented a different means of approaching the problem, and each presented benefits as well as problems. The console window allowed me not to worry about where windows are position and the what not that comes with programming in windows, while programming in windows allowed me to edit the input going into the program.

A STUDY OF GENETIC ALGORITHMS

Colin M. Kinsella

Introduction-

The study of genetic algorithms all started with a man by the name of John Holland. He invented the field of genetic algorithms in the early 1970's to mimic some of the processes observed in natural evolution into a computer algorithm.

Genetic Algorithms are based on models that optimize rules by mimicking the Darwinian Law of survival of the fittest. A set of rules are chosen by those that work the best. The weakest are discarded. In addition, two successful rules can be combined (the equivalent to genetic cross-overs) to produce offspring rules. The offspring can replace the parents, or they will be discarded if less successful than the parents. Mutation is also accomplished by randomly changing elements. Mutation and cross-over occur with low probability, as in nature.

Genetic Algorithms are actually search algorithms that use a parallel search technique based on a couple aspects of evolution. These algorithms use natural selection and reproduction to determine an optimal or near-optimal solution.

In genetic algorithms, the individual solutions are referred to as chromosomes. Chromosomes are based on the reproduction of strands of deoxyribonucleic acid(DNA). DNA is made up of cytosine, guanine, thymine, and adenine. Chromosomes are replicated during times of cell replication, which include mitosis and meiosis. For

example human cells contain 46 chromosomes. When replication occurs a sex cell is split to form two gametes that each contain 23 chromosomes. The cells formed can be either a sperm or an egg depending on the sex of the individual. The sex of an individual is determined by the X and Y chromosomes. When fertilization occurs a zygote is formed. This brings in a broad perspective because along the way many doors are opened up. Many things can occur such as random union of gametes, cross-overs, genetic recombinations, mutations, and perhaps a few other things.

All of this applies to genetic algorithms by the first chromosomes used in genetic algorithms were based on a binary encoding of the chromosomes, which is simpler than the four letter encoding system that makes up DNA(A,T,G,C). The chromosomes in genetic algorithms serve as memory for the current or interim solutions for a problem. Genetic algorithms take chromosomes and reproduce them to make a new generation of chromosomes that will hopefully be better than the last.

Genes are the individual elements of chromosomes, and chromosomes represent the current solutions in a population. In genetic algorithms, genes can be as complicated or as simple as needed to fit the problem. The terminology of genetic algorithms is based on biology, where they mimic, with the theory of evolution and natural selection as their inspiration.

The concept of evolution is the key to understanding genetic algorithms. The following is from [Davis 4] and explains the important aspects of the theory of evolution which created genetic algorithms:

1. Evolution is a process that operates on chromosomes rather than on the living beings they encode.

2. Natural Selection is the link between chromosomes and the performance of their decoded structure.
3. The process of reproduction is the point at which evolution takes place via mutation and recombination.
4. Biological evolution has no memory -- information is contained in the gene pool -- set of chromosomes carried by current individuals.

It is important to also understand that genetic algorithms do not know the type of problem they are solving. That information is only contained in the chromosome and the fitness evaluation function.

John Holland believed that very difficult problems could be solved using genetic algorithms, just as evolution has done through nature. His initial binary strings consisted of 1's and 0's, which made up the chromosomes.

These algorithms used simple logic and encoding procedures, but produced large and complicated results that could solve extremely difficult problems quickly are resourcefully. In reference to their study of genetics, Holland named these algorithms "Genetic Algorithms."

Basic Structure of a Genetic Algorithm-

Two mechanisms link a genetic algorithm to the specific problem it is trying to solve:

1. The encoding of the chromosome contains information specific to the problem.
2. An evaluation function of the fitness of a chromosome. The evaluation function acts like the environment in natural selection.

The following is a basic example of an outline of a genetic algorithm.

The Genetic Algorithm [Davis 4]

1. Initialize the population of chromosomes.
2. Evaluate each chromosome in the population.
3. Create new chromosomes by mating current chromosomes; apply *mutation* and *recombination* as the parent chromosomes mate.
4. Delete members of the population to make room for the new chromosomes.
5. Evaluate the new chromosomes and insert them into the population.
6. If time is up, stop, and return the best chromosome; if not go to step 3. It is to be hoped that an initial population of unexceptional chromosomes will improve until it produces a highly evolved solution to the problem.

Basic Parts of the Genetic Algorithm-

The basic and most important parts of a genetic algorithm are chromosome fitness evaluation, crossovers, mutations, and population deletions.

Chromosome Fitness Evaluation:

The evaluation algorithm is the crucial part of making a genetic algorithm work by finding the most optimal solution.

Crossovers:

Crossovers distinguish genetic algorithms from other optimization algorithms through their role of allowing advantageous traits to be spread throughout the population in order that the population as a whole may benefit from.

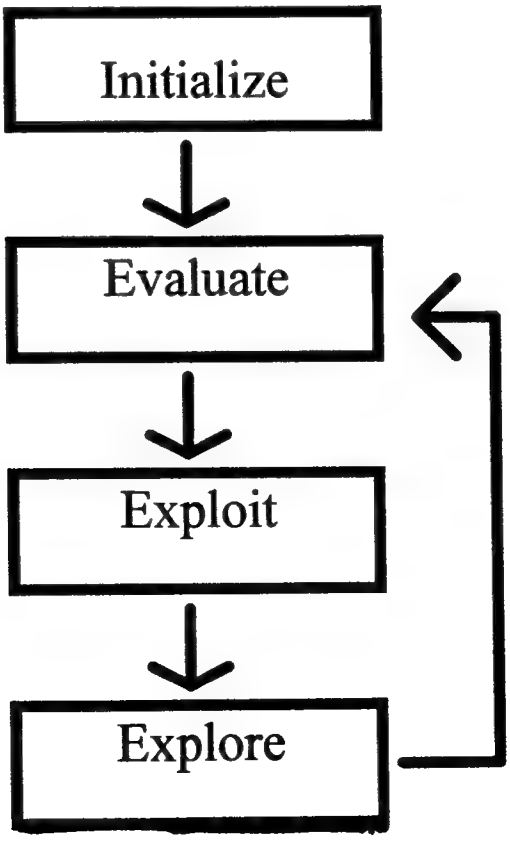
Mutations:

The purpose of mutations is to provide a divergence into a converging population. The mutation can be thought of as an error in the reproduction process. Although this is an error, it is still very important since it is the only possible way to create truly new chromosomes from old chromosomes.

Population Deletion:

This is the part of the genetic algorithm that puts new chromosomes into the existing population. This can be done in a couple of different ways. The new chromosomes can either be put into an empty population which will become the new current population, or the population can continuously be updated during the mating process, allowing new chromosomes to act like parents to the children in their own generation. The former deletion technique is called "Delete All" and the latter is called "Steady Delete."

Genetic Algorithms work by four basic steps as shown by this flow chart:



This is all started with an initial population of chromosomes that is either created randomly or by perturbing an input chromosome. The initial population should span a wide range of variable settings (Shaffer 2).

In the second step, evaluation, the fitness is computed. The goal of the fitness function is to numerically encode the performance of the chromosome. For real-world applications of optimization methods such as the genetic algorithm's choice of the fitness function is the most critical step (Shaffer 2).

The third step is the exploitation or natural selection step. In this step, the chromosomes with the largest fitness scores are placed one or more times into a mating subset in a semi-random fashion. Chromosomes with the low fitness scores are removed from the population. There are several methods for performing exploitation. One of the most common methods is the binary tournament mating subset selection method. In this mating selection method, each chromosome in the population competes for a position in the mating subset. Two chromosomes are drawn at random from the population, the chromosome with the highest fitness score is placed in the mating subset. Both chromosomes are then returned to the population and another tournament begins. This procedure will continue until the mating subset is full. A characteristic of this scheme is that the worst chromosome in the population will never be selected for inclusion in the mating subset (Shaffer 2).

The fourth step, exploration, consists of the recombination and mutation operators. Two chromosomes(parents) from the mating subset are randomly selected to be mated. The probability that these chromosomes are recombined, mated, is a user-controlled option and is usually set to a high value. If the parents are allowed to mate, a

recombination operator is employed to exchange genes between the two parents to produce two children. If they are not allowed to mate, the parents are placed into the next generation unchanged. The two most common recombination operators are the one-point and two-point crossover methods. In the one-point method, a crossover point is selected along the chromosome and the genes up to that point are swapped between the two parents. In the two-point method, two crossover points are selected and the genes between the recombination operator, is the uniform crossover method. In this crossover method, recombination is applied to the individual genes in the chromosome. If crossover is performed, the genes between the parents are swapped and if no crossover is performed the genes are left intact. This crossover method has a higher probability of producing children which are much different from their parents so the probability of recombination is usually set to a low value. The probability that a mutation will occur is another user-controlled option and is usually set to a low value so that good chromosomes are not destroyed. A mutation simply changes the value of a particular gene (Shaffer 2).

After the exploration step, the population is full of newly created chromosomes (children) and steps two through four are repeated. This process continues for a fixed number of generations.

Methodology-

From the programming aspect of making the genetic algorithm, I tried to use Object Oriented Programming. This is a style of programming where everything is broken down into simpler objects. It is centered around several major concepts: abstract data types and classes, subclasses, inheritance, and polymorphism. All of this adds to a

better readability, reliability, and lower costs encompassing the software. Using Object Oriented Programming, a program someone could have written twenty years ago will still be just as well understood now then as it was written.

Conclusion-

Although I did not finish the whole program, I learned a lot about programming in C++ and programming in windows to build an application. Object Oriented Programming could have really helped the overall program. I could not find a way to really implement the object oriented programming into writing a windows application for the genetic algorithm problem. It probably comes with practice. Over time I will be able to implement it better, and make my programs easier to read by other people and myself.

I also learned a great deal about debugging programs. I used the debugger that came with Microsoft Visual C++ 4.0. This really helped figuring out a wide variety of problems. I just had to tell the debugger what variables I wanted to watch, etc., and it would return the values at different break points that I specified while running the application. It made it a lot easier to see the mistakes I made while writing the program.

In short, I learned a lot during the course of these eight weeks that will help me choose a career in the future.

WORKS CITED

Davis, Lawrence. Handbook of Genetic Algorithms. Boston: International Thomson Computer Press, 1996.

Pinson, Lewis and Richard Wiener. An Introduction to Object Oriented Programming and C++. New York: Addison-Wesley Publishing Company, Inc., 1988.

Shaffer, Ron. Practical Guide to Genetic Algorithms. Http://chem1.nrl.navy.mil/~shaffer/practga.html.

A STUDY OF HOSTILE ELECTROMAGNETIC ENVIRONMENTS WITHIN MULTICHIP MODULES

Matthew A. Miling

Vernon-Verona-Sherrill High School
5275 Route 31
Verona, NY 13478

Final Report for:
High School Apprenticeship Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington DC

and

Rome Laboratories

August 1997

A STUDY OF HOSTILE ELECTROMAGNETIC ENVIRONMENTS WITHIN MULTICHIP MODULES

Matthew A. Miling
Vernon-Verona-Sherrill High School

Abstract

Hostile environments within multichip modules were studied. The study emphasized how electromagnetic wave propagation occurs when signals traveling along a microstrip line reach a bend or, in this case, a via. The patterns of phase and amplitude were studied at probes inserted within the microstrip simulation. The Finite Difference Time Domain Method (FDTD) was used to accurately characterize the propagating energy traveling throughout the module. Future work will involve the program we developed to investigate a microelectromechanical system (MEMS) device.

A STUDY OF HOSTILE ELECTROMAGNETIC ENVIRONMENTS WITHIN MULTICHIP MODULES

Matthew A. Miling

Introduction

As multichip modules succeed in achieving size reduction to maximize space efficiency, the effects of electromagnetic interactions at such diminutive levels are distinctively evident. The importance of observing such hostile electromagnetic environments within multichip modules becomes apparent when electronic devices, like those utilized by the United States Air Force, are constantly exposed to these conditions, as stated by Reuter, Seifert and Karle [1]. As mentioned by Tripathi, Lutz and Tripathi [2], such phenomena as crosstalk, ground bounce, multiple reflections and noise coupling all contribute to possible signal delays. Predicting the possible situations in which hostile environments occur is essential to understanding how dependable a system is under these operating conditions.

Methodology

The simulation of the propagating waves was made possible by the FD-TD method in solving the Maxwell curl equations for electric and magnetic fields. This method incorporates the use of the discretized version of the curl equations to accurately simulate propagated electromanetic energy. The fields are created, propagated and scattered within a three-dimensional space lattice with the ability to model the electrical properties of all materials; in our case, we modeled a metal layer for the microstrip, a dielectric layer, and air. A numerical gaussian source is inserted within the region and the FD-TD technique simulates the energy propagation throughout the grid including radiation penetration and scattering from modeled objects.

Maxwell's equations describe the behavior of electric and magnetic field vectors caused by signal propagation. Maxwell's equations for a source-free region are given in differential form by

$$\frac{\partial \vec{B}}{\partial t} = -\nabla \times \vec{E} - \vec{J}_m \qquad \frac{\partial \vec{D}}{\partial t} = \nabla \times \vec{H} - \vec{J}_e$$

$$\nabla \cdot \vec{B} = 0 \qquad \nabla \cdot \vec{D} = 0$$

where: $\vec{B}$: Magnetic Flux Density (Wb/m^2) $\vec{D}$: Electric Flux Density (C/m^2)

$\vec{E}$: Electric Field Intensity (V/m) $\vec{H}$: Magnetic Field (A/m)

$\vec{J}_m$: Current Density (A/m^2) $\vec{J}_e$: Current Density (A/m^2)

which completely describe the electric and magnetic fields. These expressions can be transformed into discrete forms for implementation on a digital computer according to a procedure described in detail by Taflove [3]. The update expression for the x-directd electric field is given by:

$$E_x\Big|_{i,j,k}^{n+1} = C_{a,E_x}\Big|_{i,j,k} E_x\Big|_{i,j,k}^{n} + C_{b,E_x}\Big|_{i,j,k} \cdot \left(H_z\Big|_{i,j+1/2,k}^{n+1/2} - H_z\Big|_{i,j-1/2,k}^{n+1/2} + H_y\Big|_{i,j,k-1/2}^{n+1/2} - H_y\Big|_{i,j,k+1/2}^{n+1/2} \right)$$

Electric field update coefficients for this expression are given by:

$$C_a\Big|_{i,j,k} = \left(1 - \frac{\sigma_{i,j,k}\Delta t}{2\varepsilon_{i,j,k}}\right) \Big/ \left(1 + \frac{\sigma_{i,j,k}\Delta t}{2\varepsilon_{i,j,k}}\right) \qquad C_{b_1}\Big|_{i,j,k} = \left(\frac{\Delta t}{\varepsilon_{i,j,k}\Delta_1}\right) \Big/ \left(1 + \frac{\sigma_{i,j,k}\Delta t}{2\varepsilon_{i,j,k}}\right)$$

The discretization process yields six finite difference expressions for the code to completely describe the electromagnetic fields. These can be found in [3].

Microstrip Via Simulation

In this particular simulation, the geometry was laid out for a microstrip via, where a signal passes from one level to another in the vertical injunction. The space of the simulated geometry spans a volume of 40 x 120 x 20 cells in the i, j, and k directions, respectively. The signal travels a horizontal distance of 60 cells until it travels upward 10 cells before it continues along the remaining path of 60 unit cells.

The geometry of the figure studied is a unique case involving a via, which lies as a bend in the vertical direction. Figure 1 displays the dimensions of the simulated circuit with the microstrip, via, and the dielectric layers as ground planes. Visualization of the signal wave propagation was done with the YZ view of figure 1 to obtain the best view for observing the electromagnetic distribution throughout the geometry.

Results

Visualization of the data demonstrated the wave dispersing throughout the region. The code-generated snapshots of the traveling signal and its electromagnetic fields are seen in figure 3. Brightness indicates the magnitude of the electromagnetic energy exhibited in the region. The signal after 0.04 picoseconds has entered the region along with a surrounding field of energy. The signal approaches the via continuously, as shown at the 0.10 picosecond time step, and at 0.20 picoseconds the signal finally travels into the via. Here, we see the various areas of electromagnetic field effects, including reflection of the wave off the via. At 0.25 picoseconds, we see the signal continue down the remaining microstrip in the top region, while radiation reflected from the via continues to scatter in the bottom region.

Two observation points placed in different positions within the region monitor the amplitude of the passing signal and its surrounding electromagnetic field. The data points are placed at strategic locations of interest, as shown in figure 2. The data can be plotted to produce accurate time versus amplitude information to detail the strength and persistence of the waves, measured magnetically as amperes per meter, or electrically as volts per meter, as shown in figure 4.

Conclusions

The simulation of the hostile environments provide accurate data in understanding the effects of traveling signals within multichip modules at the diminutive scales. As the relative magnitudes of the propagating waves imply, involuntary transmissions appear likely within dense electric circuitry. Thus, when considering further miniaturization of multichip modules in electronic devices, it is important to consider this internal phenomenon.

References

[1] Reuter, C., M. Seifert, T. Karle, "Finite Difference Time Domain Electromagnetic Code Validation Using An Infrared Measurement Technique," 13th Annual Review of Progress in Applied Computational Electromagnetics, Monterey, CA, vol. 1, 1997, pp.330-334.

[2] Tripathi, V.K., R.D. Lutz, A. Tripathi, "Electrical Measurement Issues in Electronic Packaging," EEP-Vol. 19-1, Advances in Electronic Packaging-1997, vol. 1, ASME 1997, pp. 739-743.

[3] Taflove, A., Computational Electrodynamics: The Finite-Difference Time-Domain Method, Artech House, Boston, MA.

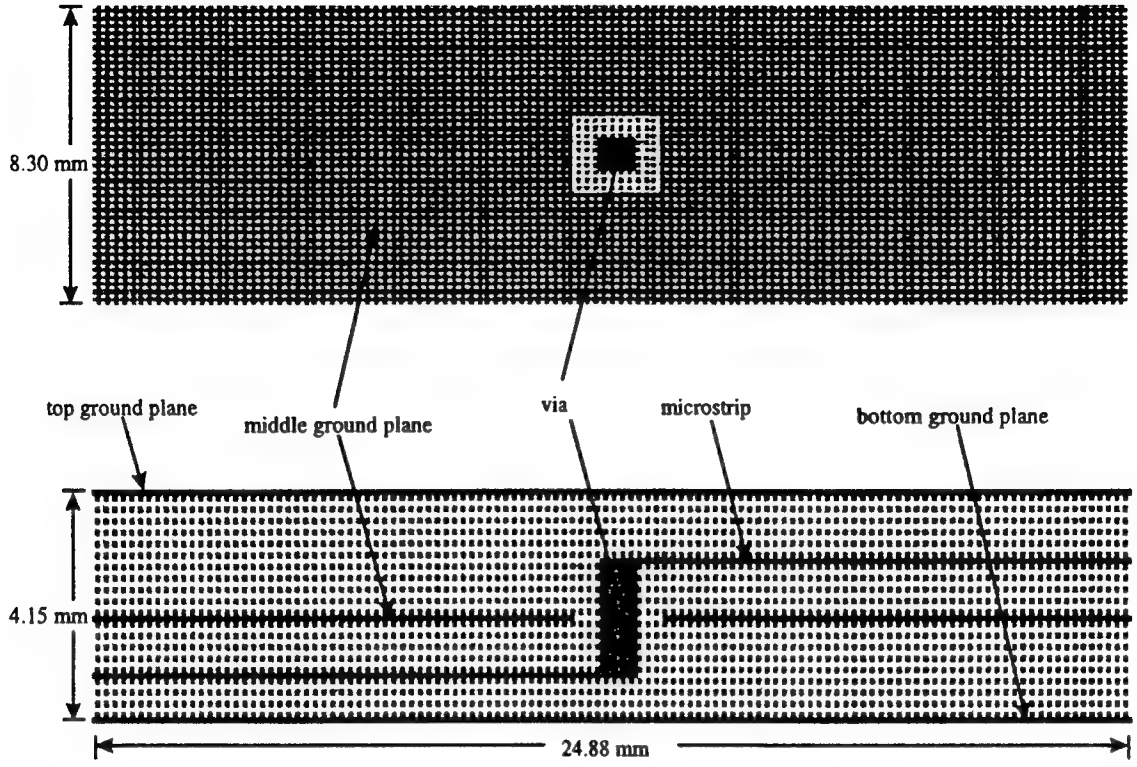


Figure 1. The top image shows the XY plane for the geometry of the simulated integrated circuit, while the bottom image shows the YZ plane.

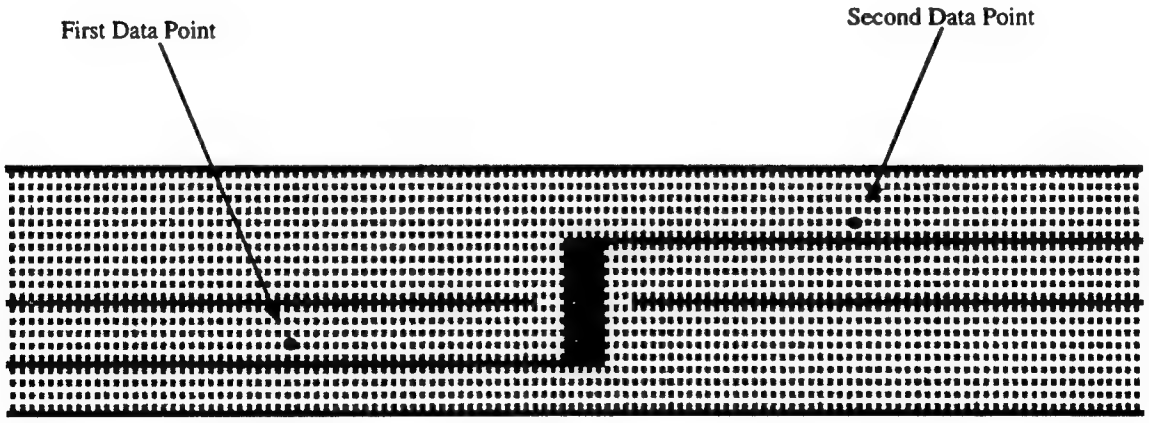


Figure 2. The observation points located in the YZ plane. The points lie directly above the microstrip, collecting data for accurate time versus amplitude plots.

signal at 0.04 picoseconds

signal at 0.10 picoseconds

signal at .20 picoseconds

signal at .25 picoseconds

Figure 3. The visualization of a signal, with electric and magnetic fields, passing through the YZ plane.

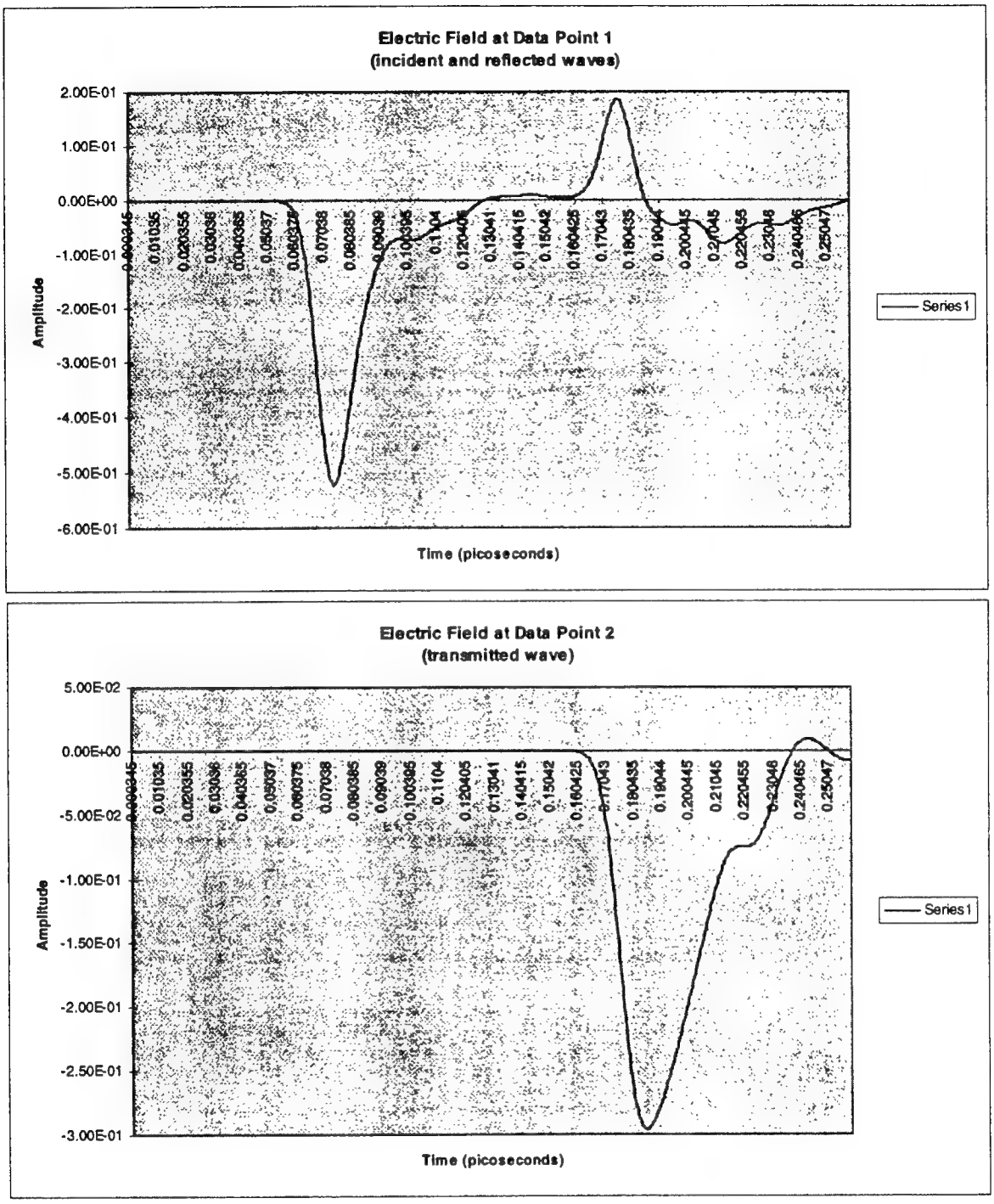


Figure 4. The amplitude versus time plots of the electric fields at the two observation points within the modeled region.

Francis Ruiz's report was not available at the time of publication.

MULTI-PARADIGMATIC PROGRAMMING: INTEGRATING PROLOG AND VISUAL BASIC

Roshan Pradip Shah

Dartmouth College
2438 Hinman
Hanover, New Hampshire 03755

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Boiling Air Force Base, DC

And

Rome Laboratory

August 1997

MULTI-PARADIGMATIC PROGRAMMING: INTEGRATING PROLOG AND VISUAL BASIC

Roshan Pradip Shah
Dartmouth College

Abstract

The use of multiple programming languages, from different programming paradigms, to cooperate on a common problem was implemented. Two separate programs were created; each profited from both Visual Basic 4.0's interface capabilities and Amzi! Prolog's logic processing. The first, WordGenius 1.0, is a flash-card program intended for vocabulary building. A random word and three not-so-random definitions are selected from the dictionary database, WordNet, using Prolog and then displayed in a Windows interface using Visual Basic. The second program, PaperCutter 1.0, computes the optimal layout for partitioning 2-dimensional space. Similar to the first, the layout is determined using Prolog and the result is displayed using Visual Basic. The use of multiple languages from different paradigms to create these programs proved to be efficient and versatile.

MULTI-PARADIGMATIC PROGRAMMING: INTREGRATING PROLOG AND VISUAL BASIC

Roshan Pradip Shah

Introduction

A technique in computer programming using multiple languages has begun to gain popularity in recent years. This technique allows programmers to determine a specific language to handle sub procedures of applications depending on strengths and weaknesses of that paradigm or language. General-purpose languages, which are traditionally used, may not always be the most efficient method to solve a problem. Obviously, doing a program's interface in the object oriented Visual Basic makes much more sense than doing it in Prolog or some other language unsuited for user interface. Thus the question is asked, why not do the entire program in Visual Basic? The answer, though, follows the same logic: another language, like Prolog, can handle the background processing better than Visual Basic. By exploiting individual languages for their strengths, programming applications become easier and faster, and the applications themselves improve in speed and capability. The future of computer science could progress at an even faster pace than it currently is by implementing this technique.

Methodology

The first program, WordGenius 1.0, was initially designed in Prolog, with a very inhibited user interface. We decided to recreate the application using Visual Basic 4.0 (VB) and integrate and the Prolog code to be the heart of the program.

The task to be completed here is to pick four words and their meanings, one of which is designated as the word to be guessed. The remaining definitions are then listed, and the user is left to differentiate the correct one from the incorrect ones. The challenge arises when we attempt to choose these other definitions. By one method, we could select three other choices from the dictionary randomly but the ease at which the correct definition could be found would contradict our initial objective: to build the user's vocabulary. Instead, we choose semantically related words to challenge the user and maintain interest. The words are chosen by Prolog from a WordNet data file. The data is arranged hierarchically in a way that allows Prolog to select the related words.

Thus, the interface was entirely under Visual Basic's control. Had it been left to Prolog, the user would have been faced with a DOS application and prompt. However, VB can create a much friendlier application in Windows format (See p. 22-7). VB also kept track of the score, percentage correct, and other minor features. Additionally, VB allowed the user to operate the program on a timer. This enabled the user to run WordGenius while running other applications, and fully maintains its purpose of teaching vocabulary by popping up with a different word.

The integration of the two languages was made possible through the use of a DLL Amzi! Prolog provided called Amzi4.bas. This universal DLL for integrating Prolog and Visual Basic allowed for the VB front end while Prolog ran in the background. At the click of the mouse a Prolog query was made and the next word and set of definitions would immediately appear. The multi-paradigmatic approach to this problem resulted in both speed and usability, with nothing

sacrificed. The same DLL, Amzi4.bas, was also integral in writing the second program, PaperCutter 1.0.

PaperCutter is a program that solves the problem of partitioning a 2-dimensional area. This program was intended to determine the optimal layout for a specialty paper cutting firm. For this reason, a constraint was placed on the problem that mandated all cuts to be made through the entire length of the sheet. The layout on page 22-13 would then be illegal, while the layout on page 22-12 would be legal.

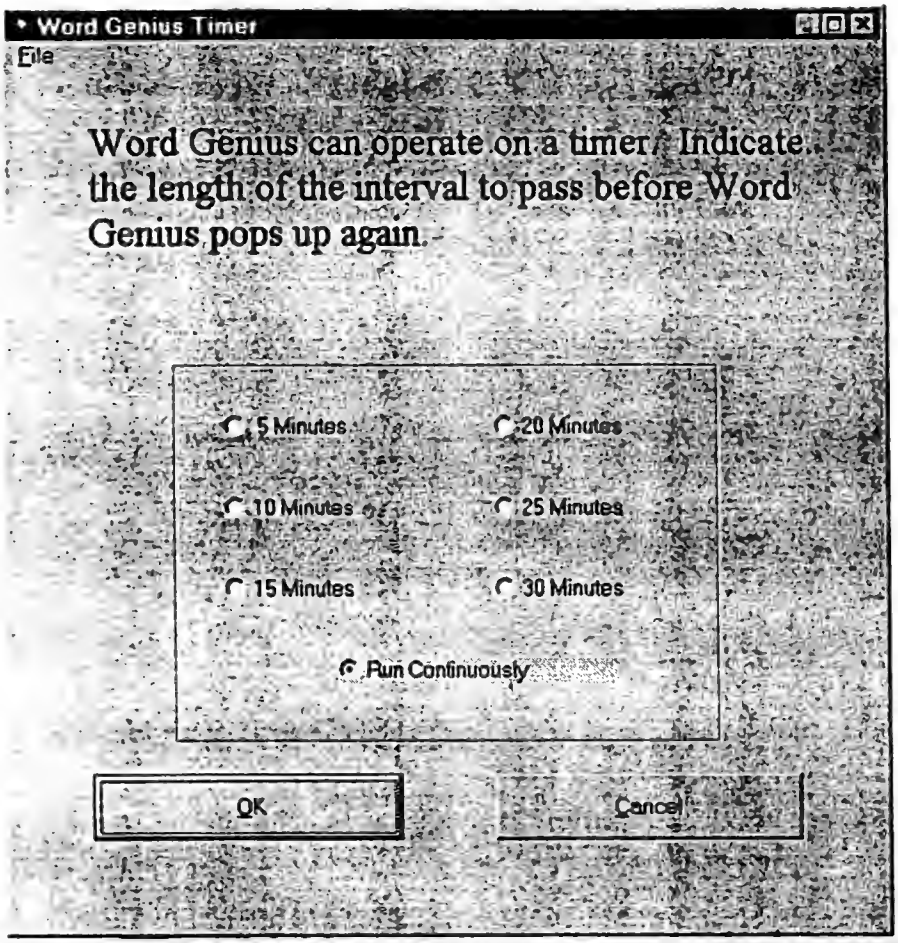
The algorithm used to determine the layout was computed using Prolog, and as before, the user interface was done in Visual Basic. However, this time the interaction between the two languages was more advanced; not only would Prolog have to send data to Visual Basic, but now it would also have to accept data as well. VB first prompts the user to enter the dimensions for the original sheet of paper and then the sheets of paper that the user wants to be placed. Once entered, the user is then asked for the length of time to let Prolog search for layouts. At that point, VB asserts this data to Prolog and waits for the results.

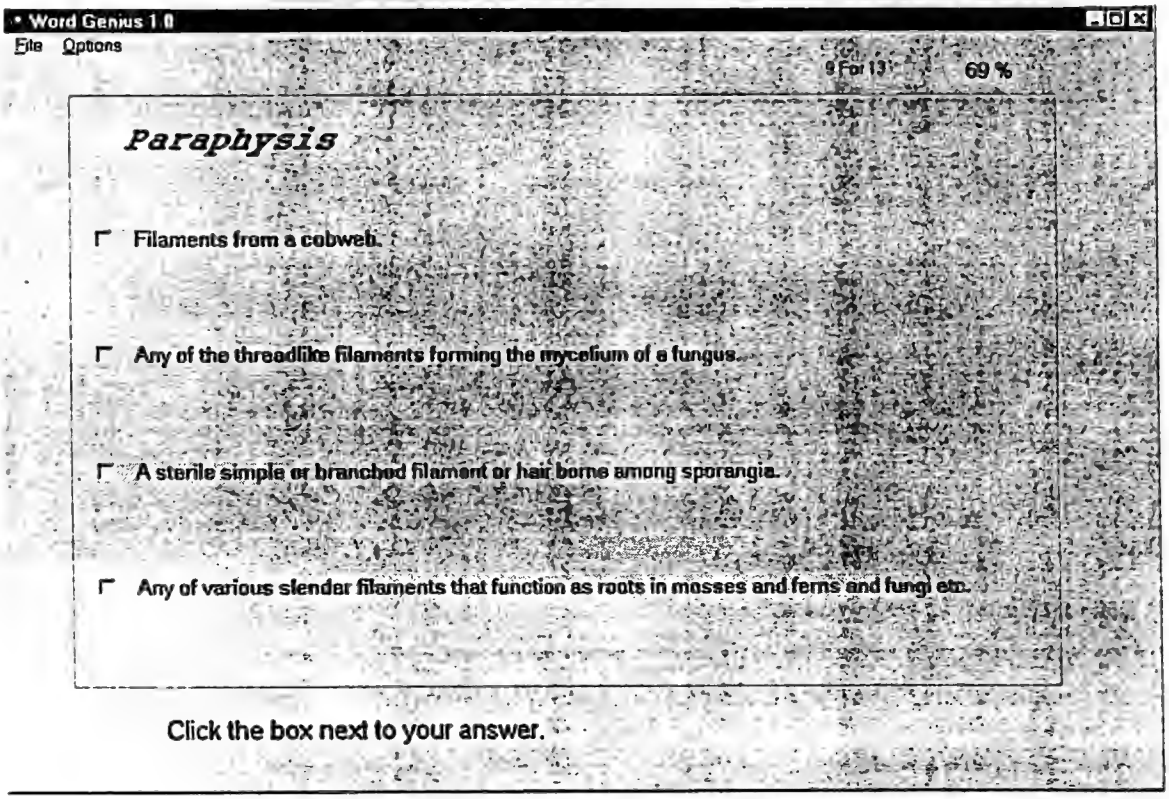
In this application, Visual Basic displays the graphical layout that Prolog produces. It also requests all the pertinent information that Prolog needs to compute the optimal layout. So again an application was created using two languages, one specializing in interface and the other specializing in logic programming.

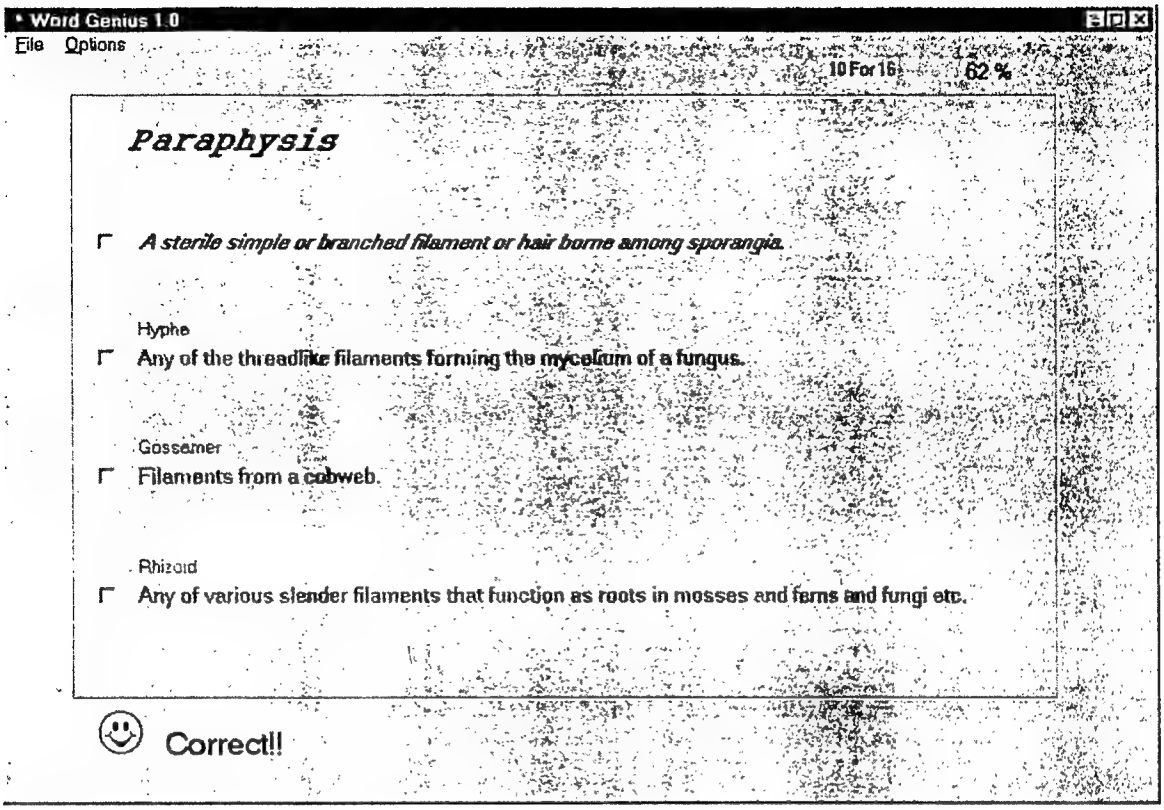
Conclusion

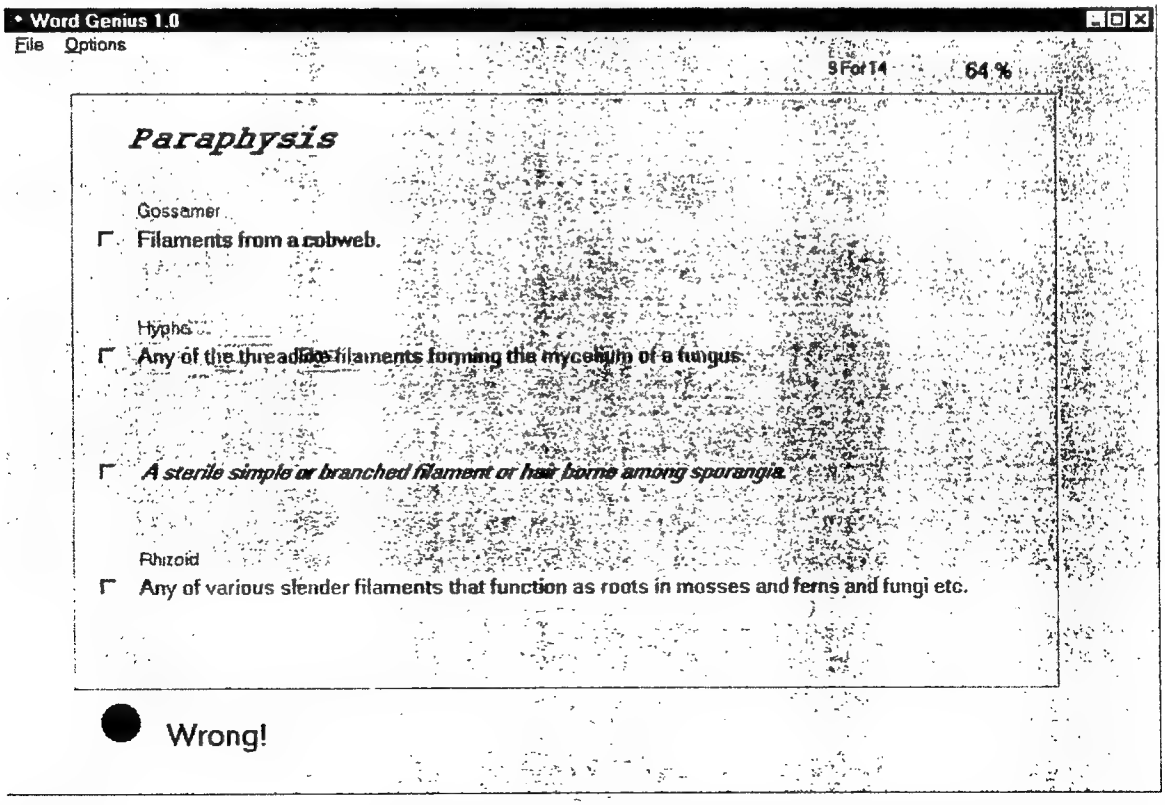
Multi-paradigmatic programming has great potential in the future of computer programming. This techniques allows programmers to exploit the advantages of particular languages to produce the best

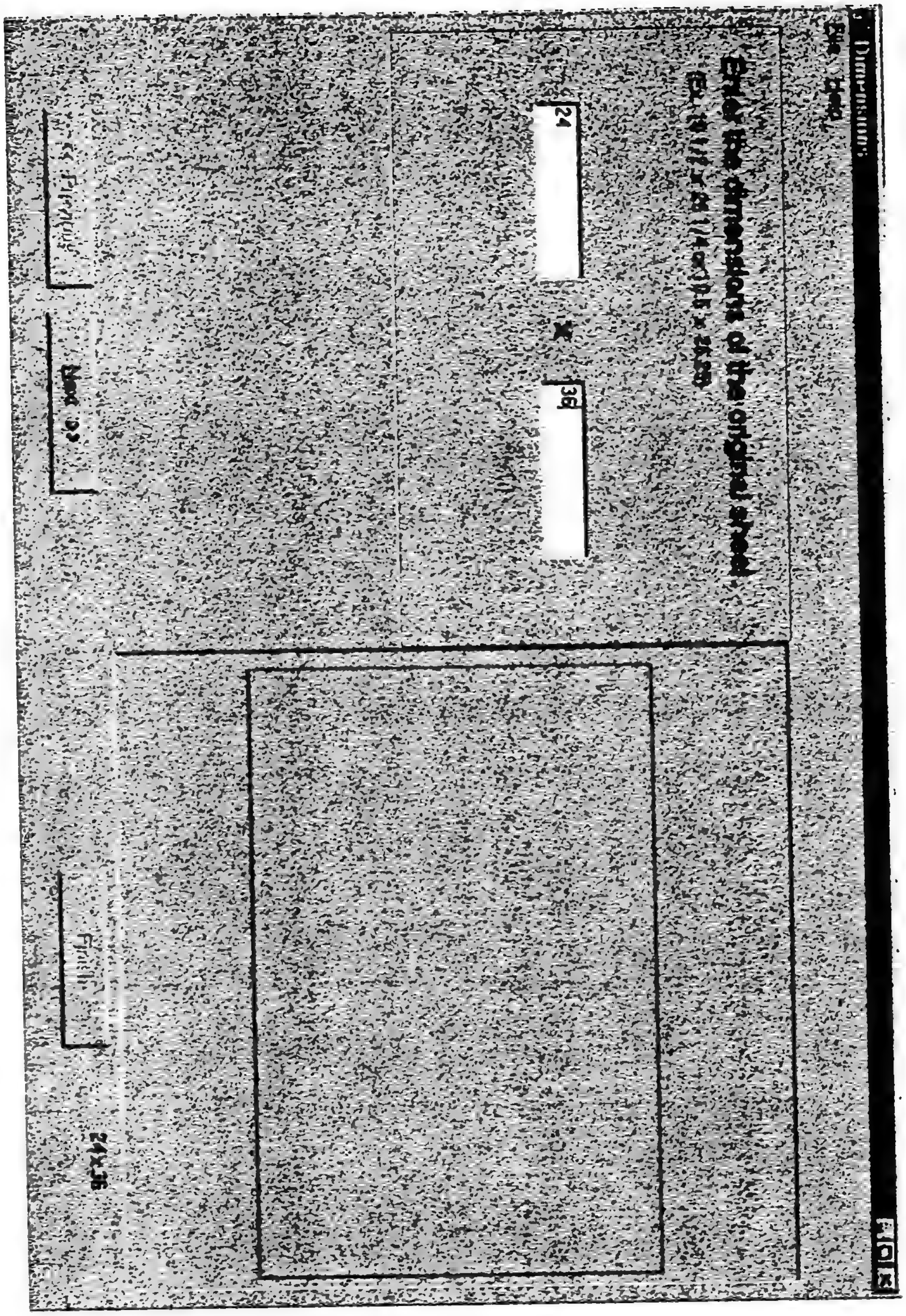
applications possible. Ultimately, we could see faster and more powerful applications. Additionally, programmers could see their programming time cut; they could give their full attention to the problem and not the different aspects of, say, presenting the interface.

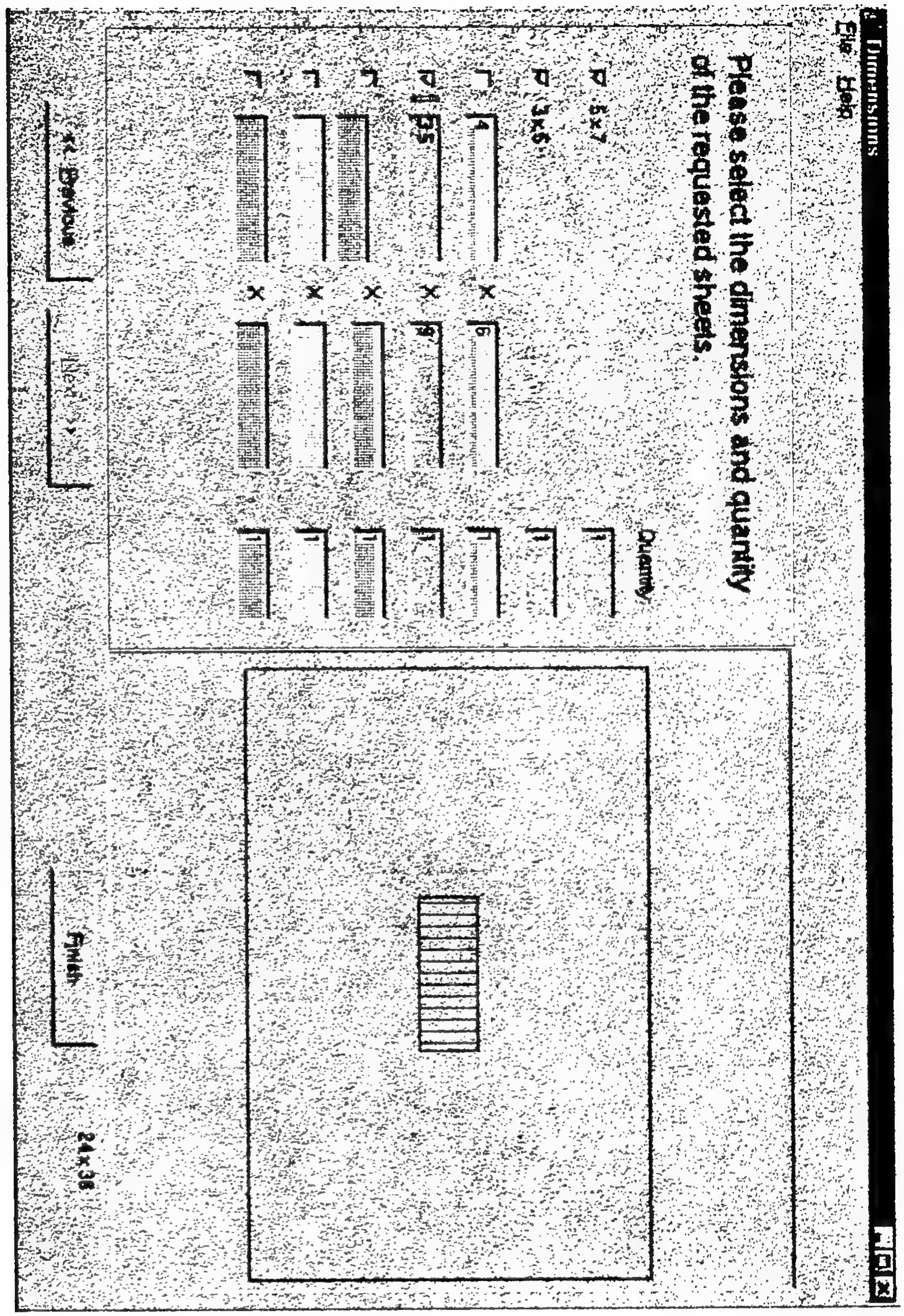

Word Genius Timer
File
Word Genius can operate on a timer. Indicate the length of the interval to pass before Word Genius pops up again.
5 Minutes
10 Minutes
15 Minutes
20 Minutes
25 Minutes
30 Minutes
Run Continuously
OK
Cancel

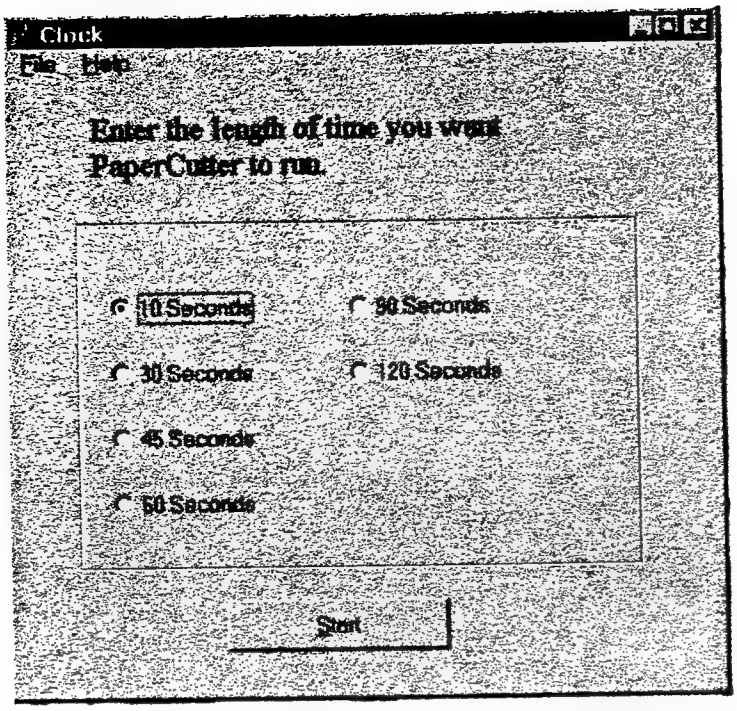

Word Genius 1.0
File Options
9 For 13
69 %
Paraphysis
Filaments from a cobweb.
Any of the threadlike filaments forming the mycelium of a fungus.
A sterile simple or branched filament or hair borne among sporangia.
Any of various slender filaments that function as roots in mosses and ferns and fungi etc.
Click the box next to your answer.

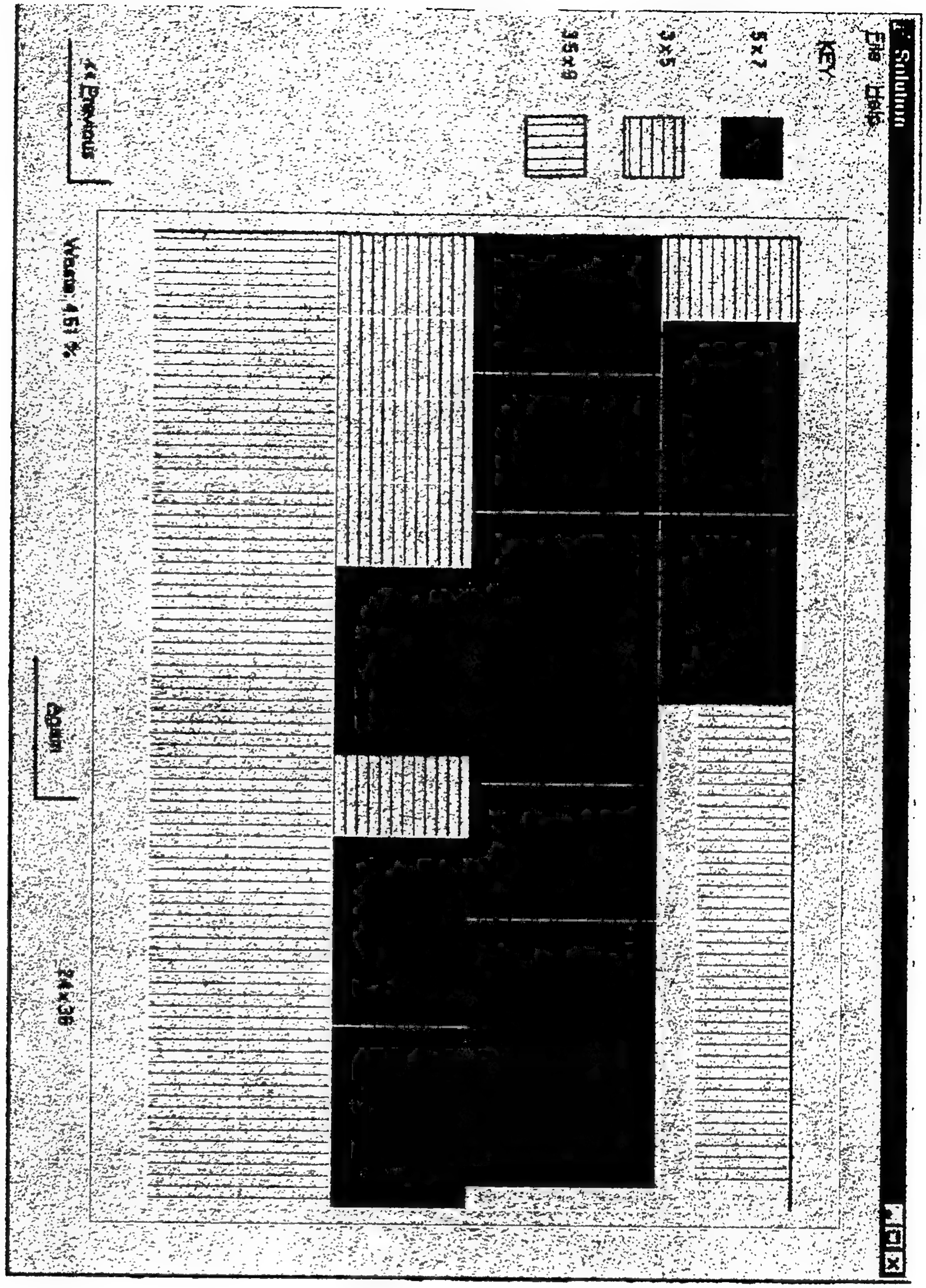

• Word Genius 1.0
File Options
10 For 16
62 %
Paraphysis
A sterile simple or branched filament or hair borne among sporangia.
Hyphe
Any of the threadlike filaments forming the mycelium of a fungus.
Gossamer
Filaments from a cobweb.
Rhizoid
Any of various slender filaments that function as roots in mosses and ferns and fungi etc.
Correct!!

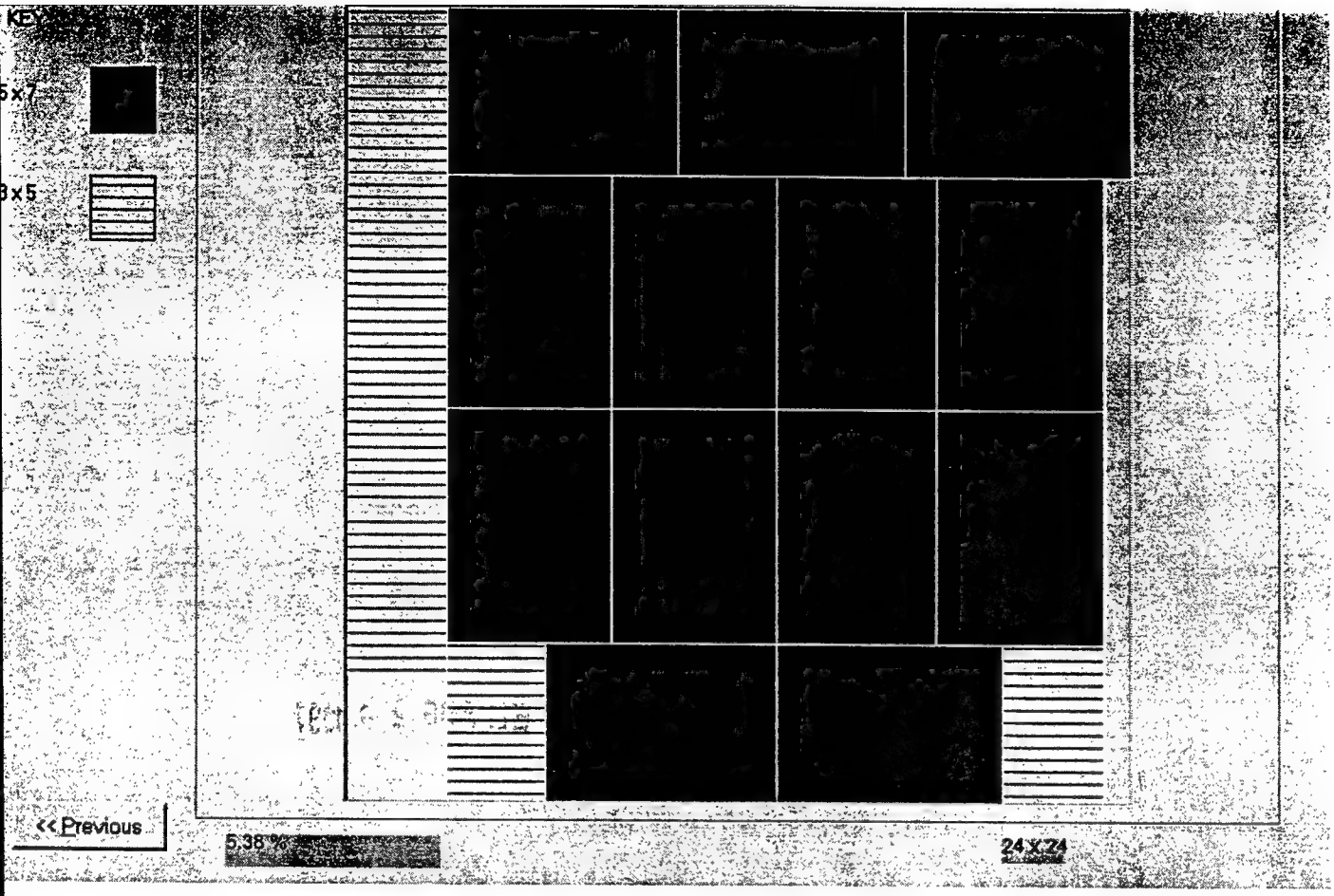

• Word Genius 1.0
File Options
9 For 14
64 %
Paraphysis
Gossamer
Filaments from a cobweb.
Hyphe
Any of the threadlike filaments forming the mycelium of a fungus.
A sterile simple or branched filament or hair borne among sporangia.
Rhizoid
Any of various slender filaments that function as roots in mosses and ferns and fungi etc.
Wrong!


Dimensions
File Help
Enter the dimensions of the original sheet
24
x
36
<< Previous
Next >>
Finish
24x36


Dimensions
File Help
Please select the dimensions and quantity
of the requested sheets.
5x7
3x5
Quantity
<< Previous
Next >>
Finish
24x36


Clock
File Help
Enter the length of time you want PaperCutter to run.
10 Seconds
30 Seconds
45 Seconds
60 Seconds
90 Seconds
120 Seconds
Start


Solution
File
Help
KEY
5 x 7
3 x 5
Again
24 x 36


5x7
3x5
<<Previous
5.38%
24x24

A STUDY OF THE APPLICATION, USES, AND PERFORMANCE OF SPREAD SPECTRUM TECHNOLOGY IN DIGITAL SIGNAL PROCESSING

Brian Tuch
RL/IRAA

Brown University
Providence, RI 02912

Final Report For:
High School Apprentice Program
Rome Laboratory

Sponsored By:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

A STUDY OF THE APPLICATION, USES, AND PERFORMANCE OF SPREAD SPECTRUM TECHNOLOGY IN DIGITAL SIGNAL PROCESSING

Brian Tuch
RL/IRAA
Brown University

Abstract

This study included exploration into Digital Signal Processing(DSP). While the focus of the experiment was on the use of spread spectrum technology, much time was spent learning basic techniques in MATLAB® and the fundamentals and mathematics behind DSP. To test the abilities of spread spectrum two scripts, one containing spread spectrum and the other not, were designed to simulate transmission and reception of a message. The results of both simulations prove the general utility of spread spectrum technology.

A STUDY OF THE APPLICATION, USES, AND PERFORMANCE OF SPREAD SPECTRUM TECHNOLOGY IN DIGITAL SIGNAL PROCESSING

Brian Tuch

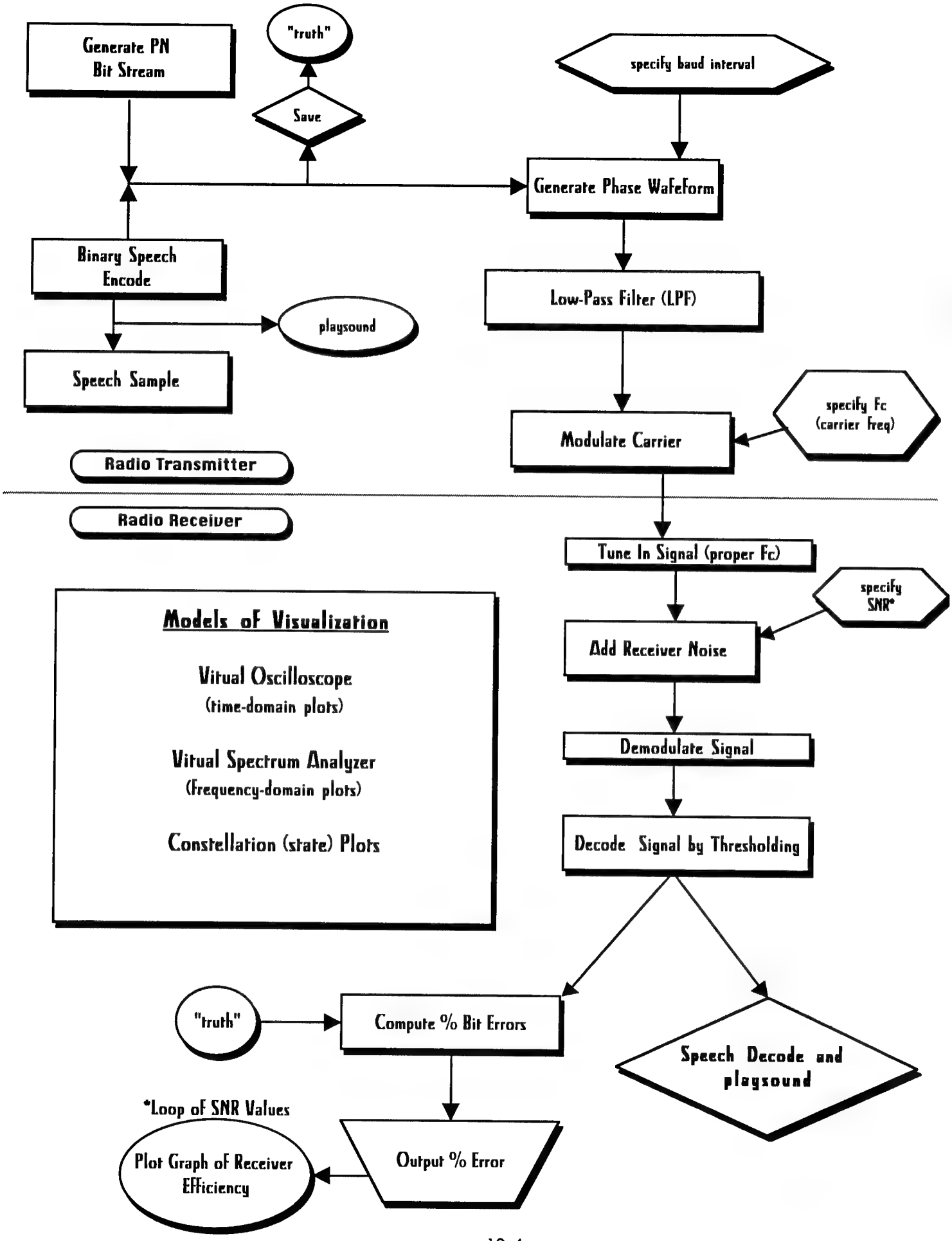
Introduction

Using Digital Signal Processing(DSP) has never been as popular as it is right now. DSP represents one of the more important accomplishments of this century, and, as proof of such, has grown applications to nearly every field of technology. Current work in the area involves speech enhancement and detection, improvement of radar systems, manipulation of images, the ever-popular Global Positioning System(GPS), and wireless communications systems. When working with DSP in communications it is often important to achieve certain objectives. These include, but are not limited to, a signal being immune to noise, having multi-user access at the same frequency and time, and the ability to send a highly secure signal. The purpose of this project is to test a very popular method , spread-spectrum by Code Division Multiple Access, of accomplishing the above, and analyze how well it provides protection from noise, protection from enemies, and ability for multiple users to access the same frequency at the same time.

Methodology

The basic methodology behind this experiment was provided by Rollie Holman, who acted as my mentor and advisor. In order to test the performance of spread spectrum it is important that a signal be sent and received. The most feasible way to carry this out was to simulate a radio transmitter and receiver in the MATLAB programming environment, thus requiring no more equipment than a personal computer. The computer used was a Macintosh 8100/110 running MATLAB 4.2c.1. Initially, this brought the experiment to a near-ideal situation.

The first step was to design a control script, one which would show the resulting performance of a receiver and transmitter not employing spread spectrum. See diagram on page 23-4. It was designed to start at the


Generate PN Bit Stream
"truth"
Save
specify baud interval
Generate Phase WaFeForm
Binary Speech Encode
playsound
Speech Sample
Low-Pass Filter (LPF)
specify Fc (carrier Freq)
Modulate Carrier
Radio Transmitter
Radio Receiver
Tune In Signal (proper Fc)
specify SNR*
Add Receiver Noise
Demodulate Signal
Decode Signal by Thresholding
Models oF Visualization
Vitual Oscilloscope
(time-domain plots)
Vitual Spectrum Analyzer
(Frequency-domain plots)
Constellation (state) Plots
"truth"
Compute % Bit Errors
Speech Decode and playsound
*Loop oF SNR Values
Plot Graph oF Receiver Efficiency
Output % Error

transmitter end by, in one case, generating a message, in the form of a random pseudonoise bit stream, held as a vector of preset length. In the other case, the bit stream is encoded from a speech sample, this simply converts the speech sample to values of 1(one) and -1(negative one). The digital speech, prior to binary conversion, can be played as audio through use of the MATLAB "playsound" algorithm. The encoded binary bit stream was then saved as a truth so, later on, it could be compared with the bit stream at the receiver end. This is how the performance of the receiver is determined. In the non-spread spectrum script the next step is to generate a phase waveform and then low-pass filter using a butterworth filter design, to smooth out the signal and make it more realistic. With this finished, the next step is to modulate the message on a carrier wave. This is done very simply by applying a carrier frequency, carrier amplitude, time vector, and the message to be transmitted to Euler's equation(below). In this case, to keep things simple, we will assume the receiver to have zero error in

$$e^{jwt} = A*e^{2*\pi*j*Fc*t + .5*\pi*j*pwf}$$

A = desired carrier amplitude
j = (-1)^.5
Fc = desired carrier frequency
t = time vector
pwf = phase waveform

tuning frequency by setting the carrier frequency(Fc) equal to zero. The result is a complex signal in transmission, and the next step is setting up the receiver.

The first step at the receiver end is to add noise, in order to better simulate a real-life system. Since the purpose of this experiment is to test performance it is best to use varying amounts of noise. To this end a signal-to-noise ratio(SNR) loop was created, which took the transmitted signal through a process of adding varying amounts of noise and computing the percent bit errors. In each run through the loop, noise was added to the signal. The signal was next demodulated, and then decoded by thresholding. The thresholding works by converting numbers greater than zero to 1 and those less than zero to -1. The output of this is compared to the truth and percentage of errors is then computed. When the SNR loop is run, it applies increasing levels of noise to the signal and outputs the percent error at each SNR to a vector. If this vector is plotted against the vector of SNR values used, a graphical view of the receiver's performance is shown.

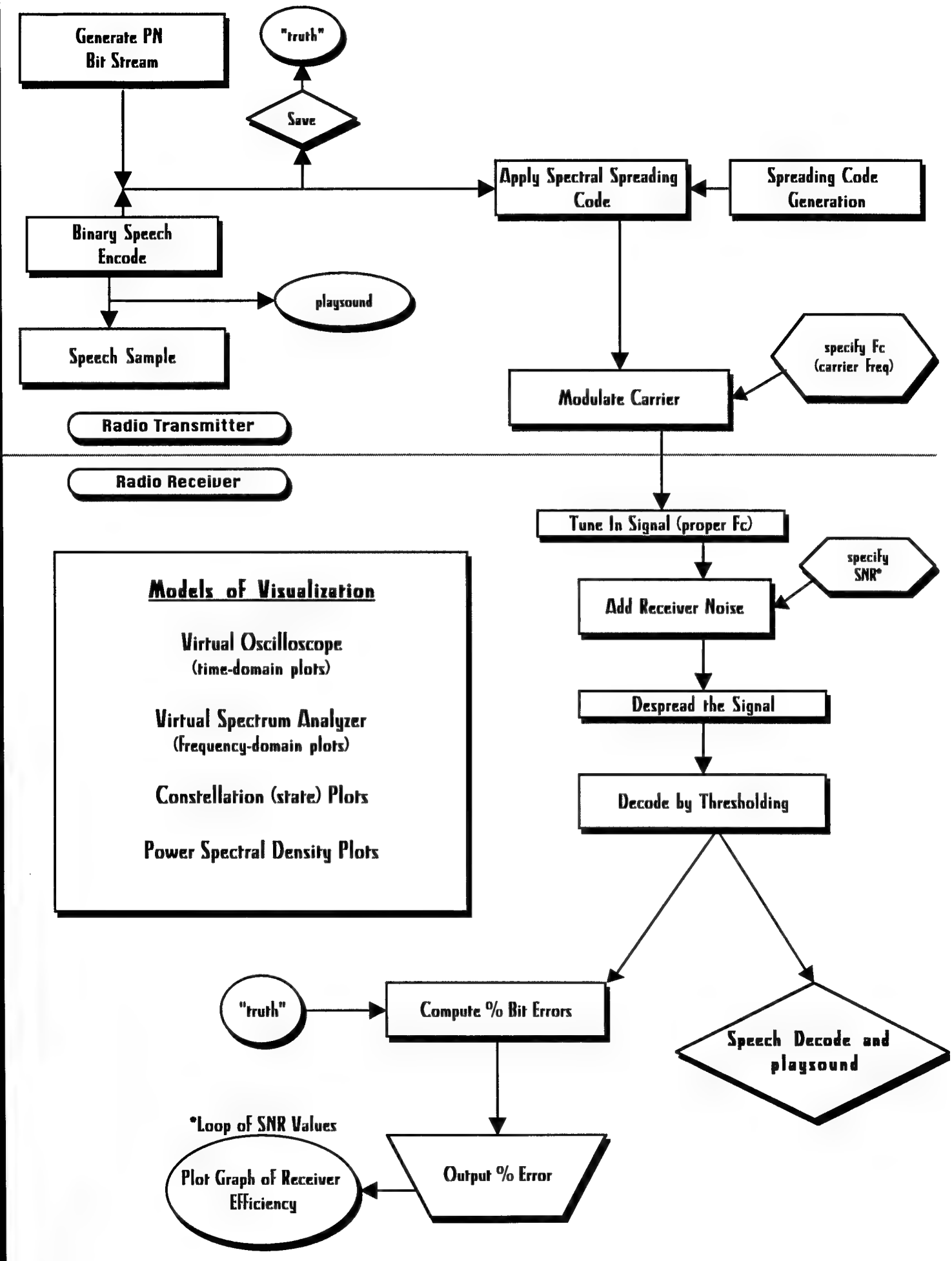
Another way of evaluating the receiver's performance, for the case of a speech sample, is to use the playsound command to hear how close the signal, on the receiver end, sounds compared to the signal on the transmitter end.

Now that a control or non-spread spectrum script had been created, the next step was to create a transmitter-receiver script utilizing some form of spread spectrum. The type of spread spectrum in this script exploited orthogonal spreading codes created by the Walsh function. The Walsh function is a popular MATLAB function for generating N sets of codes, of length N. By spreading the signal with the orthogonal codes it is hoped to achieve and demonstrate:

1. Immunity to Noise - a significant shift in the percent bit error at decreasing SNR values.
2. Multiple User Access - by applying different codes of length N, as many as N people can communicate at the same time on the same frequency.
3. Low Probability of Intercept - by spreading the signal with a code, it is highly unlikely that a person without the code could despread the signal with any accuracy.

This new MATLAB script follows much the same pattern as the old, with some functions added and others removed. See page 23-7. It, again, begins with generation of pseudonoise that is then encoded. The next step, however, is to apply the spreading codes. Here this was accomplished by block correlation(other methods, such as block convolution and match filtering, were also experimented with). Block correlation is a mathematical procedure in which each bit in the message is multiplied by the spreading code and made part of the spreaded message, thus with a code length of 64 and message size of 16 the spreaded message length becomes 1024. The spreaded message is then modulated on a desired carrier frequency and transmitted.

The next portion of the script is the receiver simulation. Once the frequency is tuned in, an SNR loop similar to that in the previous script is used. This SNR loop adds noise in the same manner, then despreads, decodes, and computes the percentage of bit errors in the received signal. Again, the output is stored in a vector that can be plotted against SNR values to graphically display the receiver performance.


Generate PN Bit Stream
"truth"
Save
Apply Spectral Spreading Code
Spreading Code Generation
Binary Speech Encode
playsound
Speech Sample
specify Fc (carrier freq)
Modulate Carrier
Radio Transmitter
Radio Receiver
Tune In Signal (proper Fc)
specify SNR*
Add Receiver Noise
Despread the Signal
Decode by Thresholding
Models of Visualization
Virtual Oscilloscope
(time-domain plots)
Virtual Spectrum Analyzer
(frequency-domain plots)
Constellation (state) Plots
Power Spectral Density Plots
"truth"
Compute % Bit Errors
Speech Decode and playsound
*Loop of SNR Values
Plot Graph of Receiver Efficiency
Output % Error

An added feature to each script is a general loop which causes the entire script, minus the setup, to be run a desired number of times, so that an average of percent bit errors can be taken. By taking an average of x runs through each script, the possibility of extreme discrepancies in percent error is diminished, and a more realistic idea of receiver performance is produced.

Other Problems Investigated:

The Code Division Multiple Access(CDMA) feature is inherent to the coding, but was tested as well. This was done simply by transmitting, receiving, and separating several spreaded messages at the same time on the same frequency and testing the accuracy of the results. The code length in the script is 64, therefore 64 messages(i.e. users) are possible.

Testing the security of the signal was done by attempting to despread the signal with an incorrect code(an enemy code). The script despreads the message by correlation of the received signal with the code, therefore, if the code is not the one used to spread the signal, the bit error percentage should be very high.

As mentioned above, there is more than one way to despread a message from the received signal. The method first used was correlation, but another technique implemented was the match filter model. While correlation works block by block to despread the signal, match filtering is continuous over the entire signal. This was done to prove the mathematical relationship between the filter and correlation. The match filter works by convolution. Convolution is simply the time reversed process of correlation, so both methods should produce equivalent results.

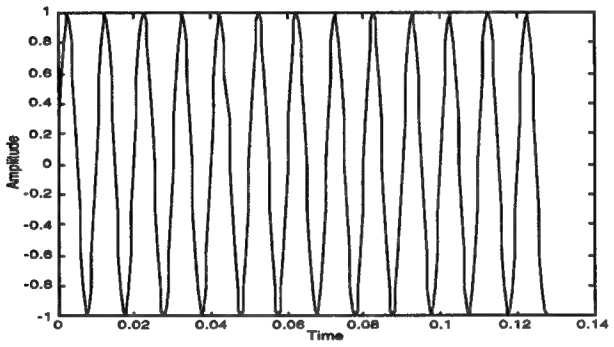
Towards the end of the project, adding a frequency tuning error was experimented with, as was an unknown start, in time, of the message. Little will be said as to the results of these trials, because both were in progress as this paper was written.

Models of Visualization

In learning DSP and attempting to understand more fully what the effect each function in the script has on the message it is helpful to employ certain visualization techniques.

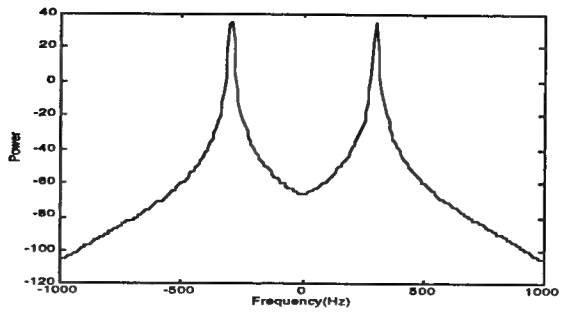
Virtual Oscilloscope - a time- domain plot

```
>>fs= 2000                    % sample rate        [Hz}
>>dt=1/fs                     % sample time interval in [sec.]
>>fn=fs/2                     % fold frequency, [Hz]
>>Ndat=2^(8)                  % data vector length
>>Nfft=256 ;                  % size/length of fft
>>Telapse=(Ndat-1)*dt;        % elapsed time in [seconds]
>>Tvector = 0:dt:Telapse ;    % time vector
>>df = fs/Nfft;               % sample frequency interval
>>Fvect = -fn:df:fn-df;       % frequency vector
>>A = 1                       % amplitude of signal
>>freq = 100
>>w = (.5*(1 - cos(2*pi*(1:Nfft)'/(Nfft+1)))).';        % hanning window

>>Sig = A *sin(2*pi*freq*Tvector);
>>plot(Tvector, Sig)
>>pause
```



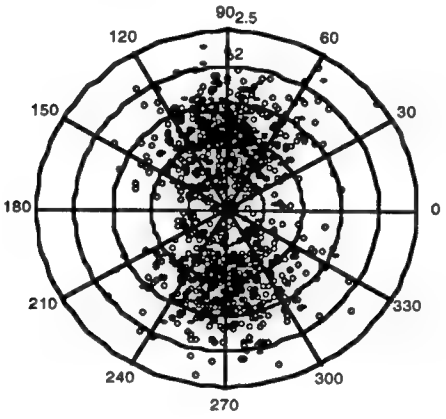
Virtual Spectrum Analyzer - a power spectral density plot - PSD

```
>>freq = 300
>>Sig = A *sin(2*pi*freq*Tvector);
>>Sig = Sig.*w;
>>SigFFTshift = fftshift(fft(Sig));
>>SpectrumDB = 20*log10(abs(SigFFTshift));
>>plot(Fvect, SpectrumDB)
```



Constellation Plots - a state plot - This is a plot of what the control script(see page 23-4 and 23-15) produced just after adding noise, at an SNR of 4, to the signal.

>> polar(angle(SignalRX), abs(SignalRX), 'bo')



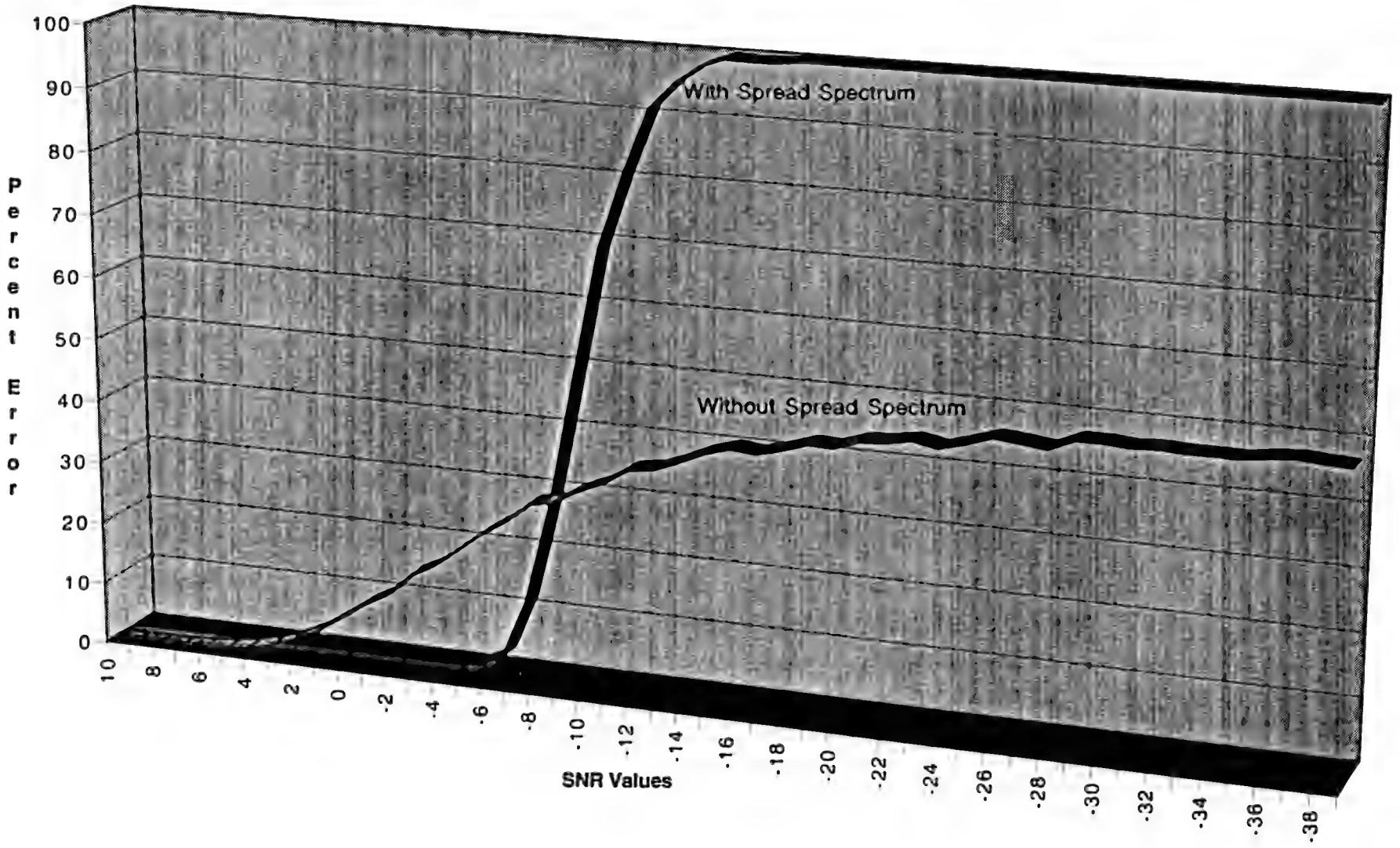
Results

The major goal of this experiment, to test the abilities of spread spectrum, was found to be entirely successful. A few minor hitches posed as threats along the way, but after altering the scripts numerous times, all were conquered, and useful results were produced.

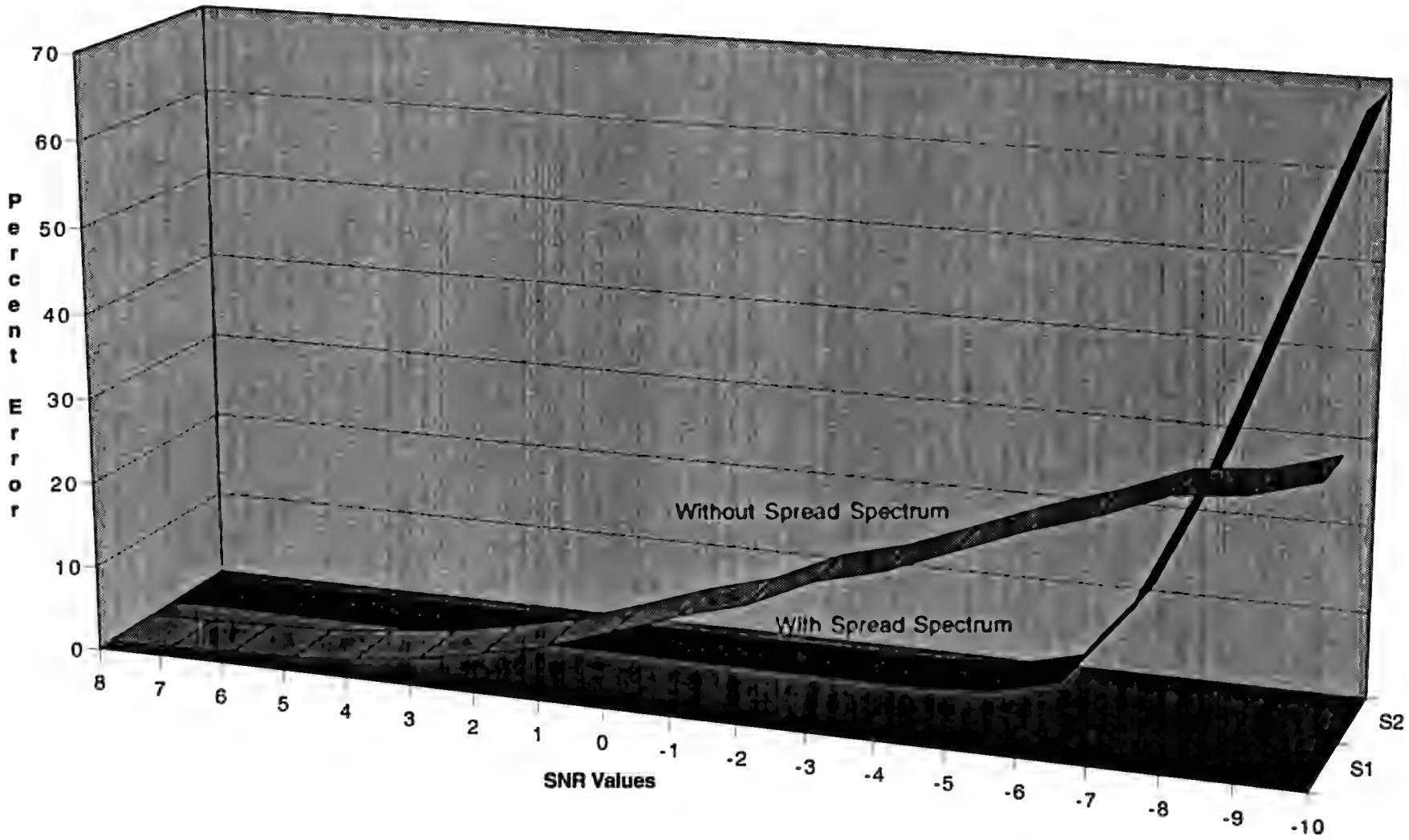
The first goal, which was to prove spread spectrum's ability to provide immunity to noise, displayed close to a 10dB gain in audibility. This improvement, over the non-spread or control case, means that a message encoded with spread spectrum can be understood when the level of noise is much higher than that of the signal, as compared to a non-spread spectrum message which begins to become inaccurate at SNRs well

SNR Value	Percent Error Without S. Spectrum	Percent Error With S. Spectrum
10	0	0
9	0	0
8	0.0195	0
7	0.1073	0
6	0.3805	0
5	0.6341	0
4	1.3951	0
3	2.2732	0
2	3.7854	0
1	5.522	0
0	8.078	0
-1	10.7902	0
-2	12.9268	0
-3	16.4585	0
-4	18.478	0
-5	21.5024	0.5
-6	24.5659	2.375
-7	27.0244	11.75
-8	30.1659	27.875
-9	30.8683	48.5
-10	33.0927	68.875
-11	34.761	80.75
-12	37.2585	91.125
-13	37.7561	95.25
-14	39.2976	98.125
-15	41.1024	99.625
-16	42.1756	99.375
-17	41.8146	99.5
-18	42.8488	100
-19	44.1756	100
-20	44.0195	100
-21	45.4049	100
-22	45.6293	100
-23	46.361	100
-24	45.6	100
-25	46.7024	100
-26	48.0683	100
-27	47.6585	100
-28	47.1122	100
-29	48.8683	100
-30	48.8585	100
-31	48.5756	100
-32	48.8195	100
-33	48.5659	100
-34	48.5951	100
-35	48.5756	100
-36	49.0927	100
-37	49.4244	100
-38	49.0927	100
-39	48.8878	100

SNR Values Used And The Resulting Percent Bit Errors In Received Message



Close Up of Percent Bit Error at SNR Values -8 though 10



above zero. These results are best depicted in a chart and graphs on page 23-11, 23-12, and 23-13 respectively. Although it is impossible to provide in this paper, the results were also demonstrated as an audio signal. In this case, we provided each script with a simple audio signal of Rollie Holman saying, "Hello great big world." This file, originally in the wav format, was converted to a binary bitstream through some of MATLAB's audio conversion functions. The bitstream was then sent through both scripts, and each script outputted a separate audio file. This was done at an SNR level of -4 because according to the above tests this SNR had high efficiency with spread spectrum and low without, see graph on 23-12. The audio files were then played and it was immediately apparent that the spreaded message was much higher in quality than the non-spreaded one. The non-spreaded audio message was not intelligible, due to noise, while the spreaded message was crystal clear, as it had been transmitted.

The test of multiple user access, or CDMA, came next. Although a code length of 64 was used, thus meaning that 64 messages could be sent out at the same time over the same frequency, only 4 were tested. CDMA as used in cellular technology sends, to each cell phone, a pilot and sync signal, so those were added as well as a second random signal. The results were a slightly lowered accuracy at low SNRs, about a 2dB loss, but all four messages were decoded successfully.

As a test of spread spectrum's security, a code other than the correct one was used to attempt despreading of the signal. The results were 0% accuracy across the board, not a single message sent between SNR values of 50 and -40 produced a correct bit. These results would make for a rather dull graph and would take up precious space, so the reader is asked to simply imagine what this would look like.

As for the different methods of despreading the code, correlation and filtering(deconvolution), both produced results that were similar, within .1% error, for each SNR. This is what was expected, due to the close mathematical relationship between correlation and convolution.

Appendix of Scripts (some of the many scripts used in the experiment)

Main Script #1

```
% DriverWfns.m  6/25/97
% (w/o Spread Spectrum)
% Written by: BBT

%%%%%%%%
%% Setup %%
%%%%%%%%

clear all                       % clear out all variables
fs = 20000;                     % sampling frequency
ffold = .5*fs                   % fold frequency
dt = 1/fs                       % time between samples
Tbaud = 5*dt                    % baud interval
Fbaud = 1/Tbaud                 % baud frequency
N = 2^10;                       % number of samples
T = (N)*dt;                     % changed this from (N - 1) to make matrix the correct length
t = 0:dt:T;                     % time vector
A = 1;                          % amplitude of transmitted carrier

Fc = 0;                         % carrier frequency - zero tuning error
Nfig = 0;                       % for plotting purposes

% Filter design
Npoles = 1;
Wn = 2*Fbaud/ffold;
[b, a] = butter(Npoles, Wn);

ErrorVectorSum = 0

for bb = 1:50                   % loop through script to increase accuracy of ErrorVector(Big Loop)

clear Message RefFrame code5 code3 WW MessSpread SigOnCarrier SnrVector ErrorVector SNR

bb

%%%%%%%%%%%%%%%%%%%%%%%
%% Generate random message bit stream.%
%%%%%%%%%%%%%%%%%%%%%%%

disp('Generate random message bit stream')
[MessLength, Message] = GenBitStream(N);

%%%%%%%%%%%%%%%%%%
%% Generate Phase Waveform %
%%%%%%%%%%%%%%%%%%

disp('Generate Phase Waveform')
PhaseWF = GenPhaseWF(MessLength, Message);

%%%%%%%%%%%
%%  Lowpass Filter  %%
%%%%%%%%%%%

PhaseWFfilt = filter(b, a, PhaseWF);

%%%%%%%%%%%%
%% Modulate Carrier  %%
%%%%%%%%%%%%

SignalTX = A*exp(j*2*pi*Fc*t + j*(pi/2)*PhaseWFfilt);

%%%%%%%%%%
%%  SNR Loop  %%
%%%%%%%%%%

clear SnrVector ErrorVector
```

```
disp('START SNR LOOP')
[SnrVector, ErrorVector] = SNRloop(SignalTX, MessLength, Message, A);

%%%%%%%%%%%%%%%%%%%%%%%%
%% Plot and Display  Error For SNR Values     %%
%%%%%%%%%%%%%%%%%%%%%%%%

ErrorVectorSum = ErrorVectorSum + ErrorVector;   % accumulate results

end                    %Big Loop

ErrorVector = ErrorVectorSum/bb;                    % divide by number of cycles(50)
[Nfig] = PercentErrorPlot(Nfig, SnrVector, ErrorVector)
```

Main Script #2

```
% driverExperBBB.m
% (with Spread Spectrum)
% BBT    7/12/97

%%%%%%%%%
% Setup
%%%%%%%%%

clear                          % clear all variables
Nfig=0;                        % for plotting purposes
fs=2;                          % sampling frequency
fn=fs/2;                       % fold frequency
dt=1/fs;                       % time between samples
Fc=       0;                   % carrier frequecy
Acarrier = 3;                  % amplitude of carrier
m=6
Ncode=2^m                      % Code / Frame length
MessLength =2^4                % length of Message to be generated
Nfft=MessLength*Ncode          % number of samples in fft (for graphing purposes)
df=fs/Nfft
Fvectr =-fn:df:fn-df;          % frequency vector (for plotting purposes)
Tvectr = 0:dt:dt*((MessLength*Ncode)-1);              % time vector

ErrorVectorSum = 0

for bb = 1:50                  % loop through script to increase accuracy of ErrorVector (Big Loop)

clear Message RefFrame code5 code3 WW MessSpread SigOnCarrier SnrVector ErrorVector SNR

bb

%%%%%%%%%%%%%
% Memory Allocation    %%
%%%%%%%%%%%%%

code5 = [zeros(Ncode,1)];
code3 = [zeros(Ncode,1)];
code1 = [zeros(Ncode,1)];
code64 = [zeros(Ncode,1)];
RefFrame = [zeros(Ncode,1)];
Message = [zeros(1, MessLength)];

%%%%%%%%%%%%%%%%
% Walsh codes generation
%%%%%%%%%%%%%%%%

disp('GENERATING WALSH CODES')
WW=WalshFunctionsGen(m);
code5 = WW(:,5).';
code3 = WW(:,3).';
code1 = WW(:,1).';
code64 = WW(:,64).';
```

```
RefFrame = code3;               % for despreading the signal
%RefFrame2 = code5;             % for testing security - an enemy code

%%%%%%%%%%%%%%
%  Message generation
%%%%%%%%%%%%%%

[Message] = GenBitStream2(MessLength);
%[Message2] = GenBitStream2(MessLength);

Mess1 = [ ];
Mess64 = [ ];

%%%%%%%%%%%%%%
% Apply spreading code
%%%%%%%%%%%%%%

MessSpread = [ ];
MessSpread2 = [ ];

for i = 1:MessLength
        MessSpread = [MessSpread Message(1,i)*code3];
%       MessSpread2 = [MessSpread2 Message2(1,i)*code5];
%       Mess1 = [Mess1 code1];
%       Mess64 = [Mess64 code64];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%
% Modulate Carrier with Spreaded Message
%%%%%%%%%%%%%%%%%%%%%%%%%%

[SigOnCarrier] = SpreadCarrierSig(MessSpread,Acarrier,Fc,fs,Tvectr );
%[SigOnCarrier2] = SpreadCarrierSig(MessSpread2,Acarrier,Fc,fs,Tvectr );
%[SigOnCarrier3] = SpreadCarrierSig(Mess1,Acarrier,Fc,fs,Tvectr );
%[SigOnCarrier4] = SpreadCarrierSig(Mess64,Acarrier,Fc,fs,Tvectr );

%%%%%%%%%%%%%%%%%%%%%%%%%%
% Start Of Receiver (SNR Loop) Add Receiver Noise
%%%%%%%%%%%%%%%%%%%%%%%%%%

SNR = 11;               % in dB

[SnrVector, ErrorVector, correl1] = SNRloop2(SNR, Acarrier, SigOnCarrier,  RefFrame, MessLength, Ncode, Message);
%SigOnCarrier2, SigOnCarrier3, SigOnCarrier4,

%%%%%%%%%%%%%
%  Percent Error Plot
%%%%%%%%%%%%%

ErrorVectorSum = ErrorVectorSum + ErrorVector;          % accumulate results

end             % end Big Loop

ErrorVector = ErrorVectorSum/bb;                % divide  by number of cycles (50)

[Nfig] = PercentErrorPlot(Nfig, SnrVector, ErrorVector)
```

SNRloop

```
function [SnrVector, ErrorVector] = SNRloop(SignalTX, MessLength, Message, A)

SNR = 11;                       % in dB

for indxSNR = 1:50

        clear Anoise NoiseWF SignalRX SIGdemod SIGdecode Error PercentError

        SNR=SNR-1;
```

```
        SnrVector(indxSNR)=SNR;
        SNRratioVOTAGE = 10^(SNR/20);
        Anoise = (A/sqrt(2))/SNRratioVOTAGE;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Add Noise
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[NoiseWF, SignalRX] = AddNoise(Anoise, SignalTX);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Signal Demodulate
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

SIGdemod = angle(SignalRX)/(pi/2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Signal Decode
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

SIGdecode = DecodeSignal(SIGdemod, MessLength);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Percent Error Calculation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[Correct] = NumCorrect(i, MessLength, SIGdecode, Message);

Error = MessLength - Correct;

PercentError = 100*(Error/MessLength);
ErrorVector(indxSNR) = PercentError;

end
```

SNRloop2

```
function [SnrVector, ErrorVector, correl1] = SNRloop2(SNR, Acarrier, SigOnCarrier, RefFrame, MessLength, Ncode,
        Message);  %RefFrame2 Messge2 SigOnCarrier2, SigOnCarrier3, SigOnCarrier4,

for indxSNR = 1:50;

        clear Anoise NoiseWF RefSig correl1 MessageRecov SignalRX  Error PercentError

        SNR=SNR-1;
        SnrVector(indxSNR)=SNR;
        SNRratioPower = 10^(SNR/10);
        Anoise = (Acarrier/sqrt(2))/(sqrt(2)*sqrt(SNRratioPower));
        SigLength = length(SigOnCarrier);
        [NoiseWF] =  AddNoise2(Anoise, SigLength);
        SignalRX = SigOnCarrier + NoiseWF; %+ SigOnCarrier2 + SigOnCarrier3 + SigOnCarrier4;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Receiver: apply  block correlation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[correl1, MessageRecov] = DeSpread(RefFrame, SignalRX, MessLength, Ncode, Message);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Percent Error Calculation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[Correct] = NumCorrect2(MessLength, MessageRecov, Message);

Error = MessLength - Correct;

PercentError = 100*(Error/MessLength);
```

```
ErrorVector(indxSNR) = PercentError;
end        %end SNR loop
```

Conclusions

While there were small problems, for the most part, the experiment was a success. The project proved that spreading a message with orthogonal codes increases security, the number of possible users, and processing gain(the lowering of percent error at various SNR values). While proving the utility of spread spectrum technology, I was able to acquire a breadth of knowledge concerning digital signal processing, and I am optimistic that what I have learned this summer will be beneficial to me in the future. DSP is a topic that many students and workers in scientific and engineering fields are required to know, and learning about it now, has given me a jump on my peers.

With high-tech communications becoming more a necessity and less a luxury, and attempts at conservation of the precious RF spectrum becoming more numerous, the implementation of spread spectrum can only grow in coming years. Over the next decade expect words like CDMA and spread spectrum to become incorporated in the layman's vocabulary.

References

Clarkson, Peter M. and Henry Stark. Signal Processing Methods for Audio, Images, and Telecommunications. London: Academic Press, 1995.

Deller, Proakis, Hansen. Discrete-Time Processing of Speech Signals. New York: Macmillan, 1993.

Kondoz, A. M. Digital Speech: Coding for Low Bit Rate Communications Systems. Chichester: John Wiley & Sons, 1995.

Krauss, Shure, and Little. Signal Processing Toolbox: For Use with MATLAB. Natick, Mass.: The MathWorks, 1994.

Lyons, Richard G. Understanding Digital Signal Processing. Reading, Mass.: Addison-Wesley, 1997.

MATLAB: Compiler. Natick, Mass.: The MathWorks, 1995.

MATLAB: Reference Guide. Natick, Mass.: The MathWorks, 1992.

Scheer, James A. and James L. Kurtz. Coherent Radar Performance Estimation. Boston: Artech, 1993.

Steiglitz, Ken. A Digital Signal Processing Primer. Menlo Park, Ca.: Addison-Wesley, 1996.

Torrieri, Don J. Principles of Military Communication Systems. Dedham, Mass.: Artech, 1983.

Thank You

- Rollie Holman
- Emily Krzysiak
- Andy Karam
- Judy Carro
- Al Sisti
- Air Force Office of Scientific Research(AFOSR)

WEB BASED COMPUTER PROGRAMMING

Brian S. Walsh

Whitesboro Central School
6000 State Route 291
Marcy, NY ZIP

Final Report For:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory, Rome NY

August 1997

WEB BASED COMPUTER PROGRAMMING

Brian S. Walsh

Whitesboro Central School

Abstract

The concept of web based computer programming is becoming increasingly important. As the popularity and use of the World Wide Web (WWW)grows, so does the need for people who can program for it. Web based computer programming entails knowing Hypertext Markup Language (HTML), and other various computer programming languages such as Perl, Java, C, etc. Text is written in such a way so that it is like a list of written instructions for the computer to perform. It is up to the programmer's tastes and preferences on how he/she would like to perform the task.

WEB BASED COMPUTER PROGRAMMING

Brian S. Walsh

Introduction

The rapid growth of the internet has increased the interest in computer programmer's whose skills include web based programming. Web based programming is based upon the Hypertext Markup Language, known as HTML. HTML is a universal, platform independent programming language. In contrast, most computer programming languages are platform dependent which restricts certain programs from being executed on certain platforms. The use of web based computing vary from stock trading, to news services, to family home pages. The possible uses of this resource is incredible. The vast amounts of information available via the internet is so overwhelming that at some point in time there will be too much available. The ability of companies to advertise using the WWW has greatly increased the speed at which this new medium has grown. Web based computer programming is what makes this new technological innovation work. Computer programmers and their imagination are the ones who will keep the internet moving at this rapid pace. My task was to make the New World Vistas Global Awareness Virtual TestBed (NWV GAVTB) information available to the public. I also needed to restrict certain information from participants outside of the NWV GAVTB community.

Methodology

In order to create a document which is understood by a computer as HTML code and not merely text, specialized formatting is required This is accomplished by placing HTML tags within the document. Also, the file must be saved and named in such a way as to be recognized by the requesting system an HTML file. For example, all HTML documents begin with an <html > tag which signifies the beginning of the HTML document. The document file must also be named as *.html or *.htm, depending upon the origin of the server (i.e. x86 platform, Unix platform, ect.).

HTML programs are basic instructions which are iterpreted by the computer. If the instructions are, in any way flawed, the computer will not be able to perform that step, and therefore fail to correctly translate the code. For my WWW pages, I programmed in HTML and Perl. I incorporated Perl when I

needed to perform an interactive script, and used HTML coding for the static informational displays and graphics. My task was to create web pages in support of the Air Force Research Laboratory's New World Vistas Global Awareness Virtual TestBed (NWV GAVTB). This program researches the concept of Global Awareness technologies and their impact on the warfighter. Additionally, I created web pages to support the FY97 Air Force Office of Scientific Research 6.1 Program Review, which allows attendees 24 hour access to billeting and meeting information. All pages created utilized formated text, images to include animation. Additionally, File Trasfer Protocol (FTP) downloading and intereactive messages capabilities were also incorporated.

My first page was the home page for the GAVTB. It contains Lt. Michael Goeringer's e-mail link, a public "whiteboard" page to allow for public comments and suggestions, a secure "whiteboard" page for GAVTB program participants, FTP access for members, and multiple informaitonal links to other GAVTB pages and Rome Laboratory initiative web sites. The two "whiteboard" pages were designed to allow for opinions, questions, and comments concering the GAVTB. They were constructed to allow an individuals comments to be dynamically stored, allowing for immediate access following the posting of the message. The whiteboard also allows replies to the posted messages. I also set up a background program that removes messages that are no longer needed. The final set of pages I created was for the Air Force Office of Scientific Research (AFOSR) 6.1 Program Review. The first page was an overview containing the basic information on the program review, reservations, and accommodations. I created two links containing the directions from Syracuse Hancock International Airport to the Beeches and Paul Revere Lodge in Rome, NY, and from the Beeches and Paul Revere Lodge to Rome Laboratories.

The GAVTB home page was created in HTML. I created the title by using the <title> tag which allows you to put a title on your page. I then began the body of the page by inputting a <body> tag. In the body tag you can specify what color you want the background to be or if you want it to be an image. You can also specify what color you want the text to be, or what color links, visited link, and activated links are. After the body is when the real programming comes into play. First I aligned two images, one to the left and one to the right. The one on the right is an animated image. The images were inserted using the <img

src="image.gif"> tag. I then placed a header in between the images. I accomplished this by using the <h1>tag. I then placed a bar to separate my heading from the text. Next, I put a bar image to separate my headers from the text. and then placed a paragraph into the page by using a tag which designates the beginning of a new paragraph. To designate the end of a paragraph I used the . The </ > denotes the end of a certain instruction. For example, in order to end the document you must use a </html> tag. I created the links to the other pages. I also placed these links into a list by using a <ul> tag followed by a <li> tag. Then the tags were created using the <a href="somewhere.html"> tag. This is an instruction for the computer to go to the page referenced in the tag, in this case somewhere.html. I then instructed the computer to stop performing the tasks defined in the HTML document by using a </body> and a </html> tag. The </body> tells the computer that the body of the web page is complete and the </html> tag tells the computer that the whole web page is complete. For example, the home page for my project was named index.html, and here is what it looked like when it was completed:

```
<DOCTYPE HTML PUBLIC"-//IETF//DTD HTML 2.0//EN">
<MHTML>
<HEAD>
<TITLE>NEW WORLD VISTAS GLOBAL AWARENESS VIRTUAL
TESTBED</TITLE>
</HEAD>
<BODY background="bg.jpg">
<IMG SRC="earthani.gif" align=right>
<h1 align=center><font color=#ff0000>N</font>EW <font color=#ff0000>W</font>ORLD <font
color=#ff0000>V</font>ISTAS</h1>
<H2 align=center><font color=#ff0000>G</font>LOBAL <font color=#ff0000>A</font>WARENESS
<font color=#ff0000>V</font>IRTUAL <font color=#ff0000>T</font>EST<font
color=#ff0000>B</font>ED</H2><br><br><br><br>
```

The Global Awareness Virtual TestBed will "Implement a geographically distributed environment representing the Global Information Base (GIB), providing tools for creating, exploring, and demonstrating Global Awareness (GA) applications, concepts, and architectures."- Major Steve Matechik

</P>

<P>For more information: </P>

<UL>

<LI><A HREF="info.html"> GAVTB Effort</A>

<LI><A HREF="research.html">On-Going Research</A>

<LI><A HREF="paper.html"> Point Paper</A>

<LI><A HREF="comments.html">Researcher's Comments</A>

<LI><A HREF="wwwboard/wwwboard.html"> GAVTB Guestbook</A>-Leave your comments and suggestions here.

<LI><A HREF="secure/down.html"> GAVTB File Archive</A><img src="lock.gif">

<LI><A HREF="secure/pprp/wwwboard.html">Internal GAVTB Participants Response Page</A><img src="lock.gif">

<LI><IMG align=center SRC="tiger.gif"><A HREF="mailto: goeringerm@rl.af.mil">Contact Lt Michael Goeringer</A>

</UL>

AFOSR 6.1 Program Review

ISF Home Page

</BODY>

</HTML>

I also had the chance to do some ethernet wiring for the local area network (LAN) in our office. I

connected two unix based computer systems to the Rome Laboratory Intelligence Support Facility network via an SMC EtherEZ hub. I was required to build and route over 200 feet of 10 base-T cable. Of the two possibe wiring schemes, I used the AT&T wiring scheme as opposed to the TIA wiring scheme to conform with the lab standard. Following the installation, both system performed flawlessly.

Conclusion

The eight week tour was very informative, and provided me the opportunity to gain experience in various fields of computer programming and in various computer environments. I was able to work with Unix, PC, Silicon Graphics, and Macintosh based systems. I operated in SunOS5, Irix 6.3 and 6.4, MacOS, Windows NT, and Windows 95. I also programmed in various computer languages to include HTML, Perl, and some basic Java. This experience was very rewarding, and taught me many valuable lessons. Attached are selected screen shots of the web pages I created, but they are best experienced by visiting:

http://www.isf.rl.af.mil/GAVTB

NEW WORLD VISTAS

GLOBAL AWARENESS VIRTUAL TESTBED

The Global Awareness Virtual TestBed will "Implement a geographically distributed environment representing the Global Information Base (GIB), providing tools for creating, exploring, and demonstrating Global Awareness (GA) applications, concepts, and architectures."- Major Steve Matechik

For more information:

- What is Global Awareness?
- GAVTB Effort
- On-Going Research
- Point Paper
- Researcher's Comments
- GAVTB Guestbook-Leave your comments and suggestions here.
- GAVTB File Archive
- Internal GAVTB Participants Response Page

- Contact Lt Michael Goeringer

AFOSR 6.1 Program Review

ISF Home Page

Rome Laboratory Home Page

AFOSR 6.1 Program Review Registration

This years AFOSR 6.1 Program Review will be held at Rome Laboratory, Rome, N.Y. on the 10th and 11th of September.

To Register for this program review please click here. Please include the following information:

- Name
- E-mail Address
- Phone Number
- Comments

The Beeches & Paul Revere Lodge of Rome has a special room rate of $58.43 which includes tax and a full breakfast. In order to recieve the special rate you must contact The Beeches & Paul Revere Lodge at (315) 336-1775 and say that you are part of the Rome Lab Program Review. The hotel would like reservations to be made by 26 August. If not, reservations must be made by 2 September.

Directions:

- Syracuse Hancock International Airport to The Beeches/Paul Revere Lodge
- The Beeches/Paul Revere Lodge to Rome Laboratory

If you have any further questions please contact 2Lt Michael Goeringer.

Please check this page periodically for updates and additions.

REPRODUCING THE COPPER/GOLD EUTECTIC CURVE USING COMPUTER SIMULATIONS

David A. Young

Rome Free Academy
500 Turin St.
Rome, NY 13440

Final Report for:
High School Apprentice Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, DC

and

Rome Laboratory

August 1997

REPRODUCING THE COPPER/GOLD EUTECTIC CURVE USING COMPUTER SIMULATIONS

David A. Young
Rome Free Academy

Abstract

Dr. Helbig and I were interested in reproducing the Copper/Gold eutectic curve using only computer simulations. To accomplish this we used a computer program that simulates atoms' reactions to assumed interatomic forces and starting conditions. With this program we studied three different 1289-atom Wulff polyhedron systems (Copper, Gold, and a Copper/Gold eutectic alloy) to try to find the temperature at which each system melts. We used several different techniques to discover the most efficient way of determining the melting temperature. After weeks of trial and error with different methods we concluded that the method of increasing the temperature of the system very slowly while monitoring the root mean square displacement of the atoms from their original positions was the most effective way to determine the melting points in these systems. We experimented with this method to determine the surface melting temperature of these systems. We chose surface melting because the high temperatures needed for bulk melting caused evaporation. We determined that the surface melting temperature of a Copper/Gold polyhedron might be slightly below that of a pure copper system, although the temperatures of such small systems are intrinsically ill-defined.

REPRODUCING THE COPPER/GOLD EUTECTIC CURVE

USING COMPUTER SIMULATIONS

David A. Young

Introduction

Over the last 50 years the Gold/Silicon (Au/Si) eutectic curve has been of growing interest in the scientific community because of it's importance to the microelectronics fabrication industry. For example, old transistors still in use in critical applications employed Au/Si wire bonds with unknown lifetimes. Unfortunately the Gold/Silicon eutectic curve is very difficult to simulate due to the unknown interatomic forces between silicon and metal atoms. We chose to study the Gold/Copper system which is simpler because of the better understood interatomic forces between Gold and Copper, and tried to simulate the thermal behavior of that system as a precursor to work on the more interesting Gold/Silicon eutectic alloy. A eutectic curve plots the melting temperature versus the relative concentrations of the substances. The characteristics of such a plot is that these substances melt at lower temperatures when combined than when pure. While there is a thermodynamic explanation for this effect, no explanation at the atomic level is known to us. Our plan was to take various concentrations of gold and copper and determine the melting point of each. Because of time constraints we were only able to reproduce data on three systems, the pure copper, the pure gold, and a 50/50 split of copper and gold. Furthermore, complications limited our studies to surface melting. In this project various equipment and methods were used to determine the most efficient way of finding the surface melting temperature of these systems.

Equipment

Over the summer I used several different computers and programs. My main machine was a 64MB, 166MHz Pentium computer (It was later upgraded to a 200MHz machine). I also had several secondary computers at my disposal. They ranged from a 200MHz Pentium pro to a 33MHz 486. These computers displayed the vast superiority in today's computers over the older models.

The main computer program I used was called Mdem[1]. Mdem used user-supplied initial conditions and assumed interatomic forces between atoms to simulate an atomic system's reaction to these conditions.

This program allows the user to specify the positions and velocities of the atoms, and the temperature of the system. Mdem can also subject the atoms to a viscous drag. This function allows the user to decrease the energy of the system while the atoms are evolving. I also used some secondary programs to analyze the data from Mdem. One such program called Rasmol[2] took the exact positions of the atoms from Mdem and displayed them as a picture (see Figures 1 and 2) that could be rotated in 3 dimensions on the monitor or printed as a picture. Another program used for analyzing Mdem data is Mathcad. Mathcad was used to graph the RMSD and temperature of the Mdem output. Another program written in BASIC by Dr. Helbig and myself was RMSD.BAS. It took the RMSD of every atom from its starting position (a perfect Wulff polyhedron at 0 K). This was the main program that told us if the system had melted. This program took the RMSD of any output file. Since each output file was over 120 KB this would have caused a memory problem. To solve this Dr. Helbig wrote the RMSD function into the Mdem code. The result was a continuous record of RMSD throughout the whole run and it saved on computer memory and an enormous amount of data processing.

Methodology

There were many ways that we thought about reproducing the Au/Cu eutectic curve. The first problem that arose was how big to make the atom clusters. Dr. Helbig came across a paper written by S. Valkealahti and M. Manninen[3] (V&M) entitled Melting of Copper Clusters. We decided that their research on the 1289-atom copper clusters could be a good reference to our work. For that reason we chose to use the 1289-atom Wulff polyhedron. After choosing the cluster one of these such clusters had to be created to run experiments on. Dr. Helbig took an Octahedron of the appropriate size and eliminated the excess pyramids with a BASIC program he wrote. The result was a 1289-atom Copper Wulff polyhedron. (Fig.1) We later adapted the program to make a 1289-atom Copper/Gold Wulff polyhedron. (Fig.2) As you can see in Fig.2 the Wulff polyhedron is a symmetric shape consisting of squares and hexagons. We chose to run our first experiments with the 1289-atom Copper cluster. We did this because it would take less time for the systems to evolve due to the fact that copper atoms are lighter than the gold. Also comparing these findings on the copper system with the previous work by V&M would test our method of simulating with

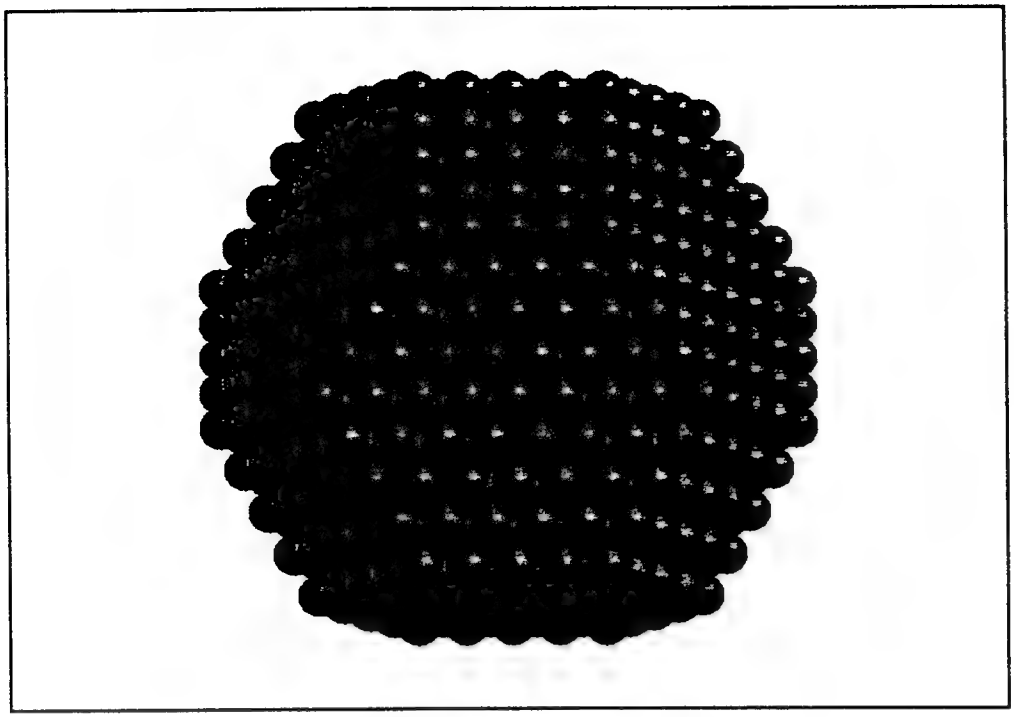


Fig.1 1289-Atom Copper Wulff Polyhedron

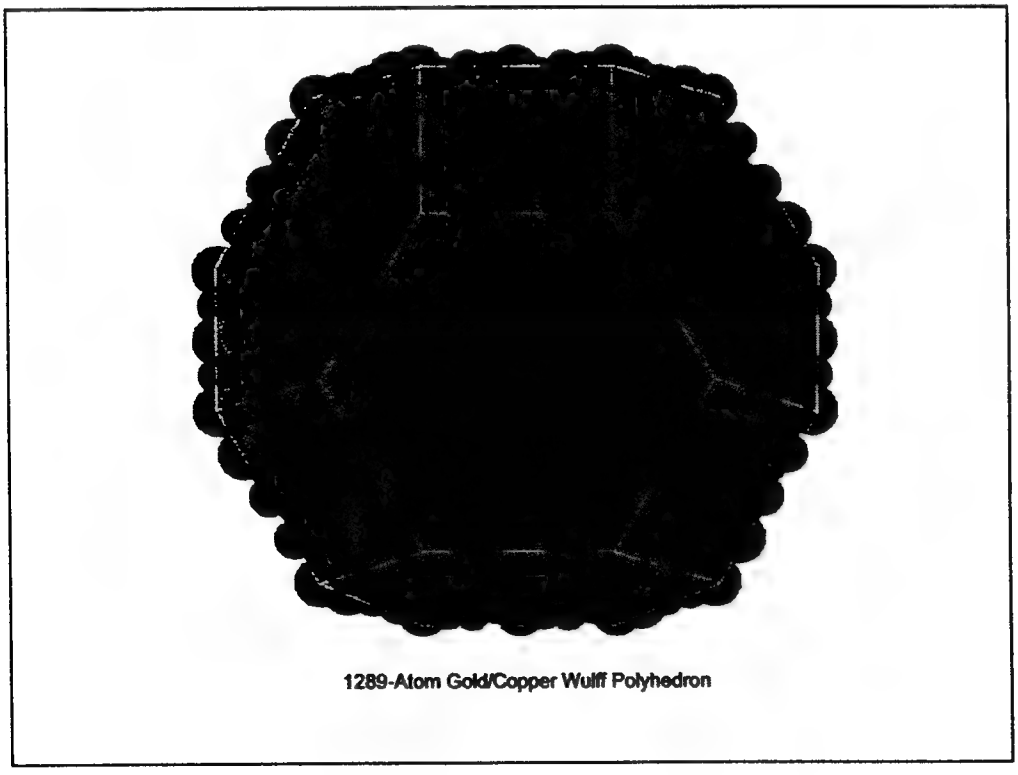


Fig.2

interatomic forces. Our first notion on how to determine the melting point of the system was to ramp the copper system up from a temperature of 0 K and print out intermediate starting files. We would take each of these files and run them until we were positive the system was melted using the RMSD program. The idea was to put a bracket around the melting point and to keep producing runs until we got to our assumed precision limit of ± 60K (this limit was implied because the temperature of small number of atoms is subject to relatively large fluctuations). This method proved impractical because the closer we got to the melting temperature the longer each run took. The final run we did with this method was a 1,000,000 integration step run at 1240 K that took three days before any melting was evident. It was obvious that this was not the best way to handle this problem.

The second method we came up with was to make a graph of the points at which each system melted. The melting temperature would then appear as a asymptote to the graph. This method also proved impractical since the graph points had large error bars. This would lead to large variations in the melting points.

The third and probably best way was patterned by V&M. We slowly increased the temperature of the systems (0.0025 K per step) and noted where the RMSD deviation "jumped" above its normal state. We couldn't take such measurements with the RMSD program we were currently using so Dr. Helbig wrote the RMSD function into the Mdem code. We also had a problem with determining the bulk melting temperature because of the evaporation caused by the high temperatures needed to bulk melt.(Fig.3)

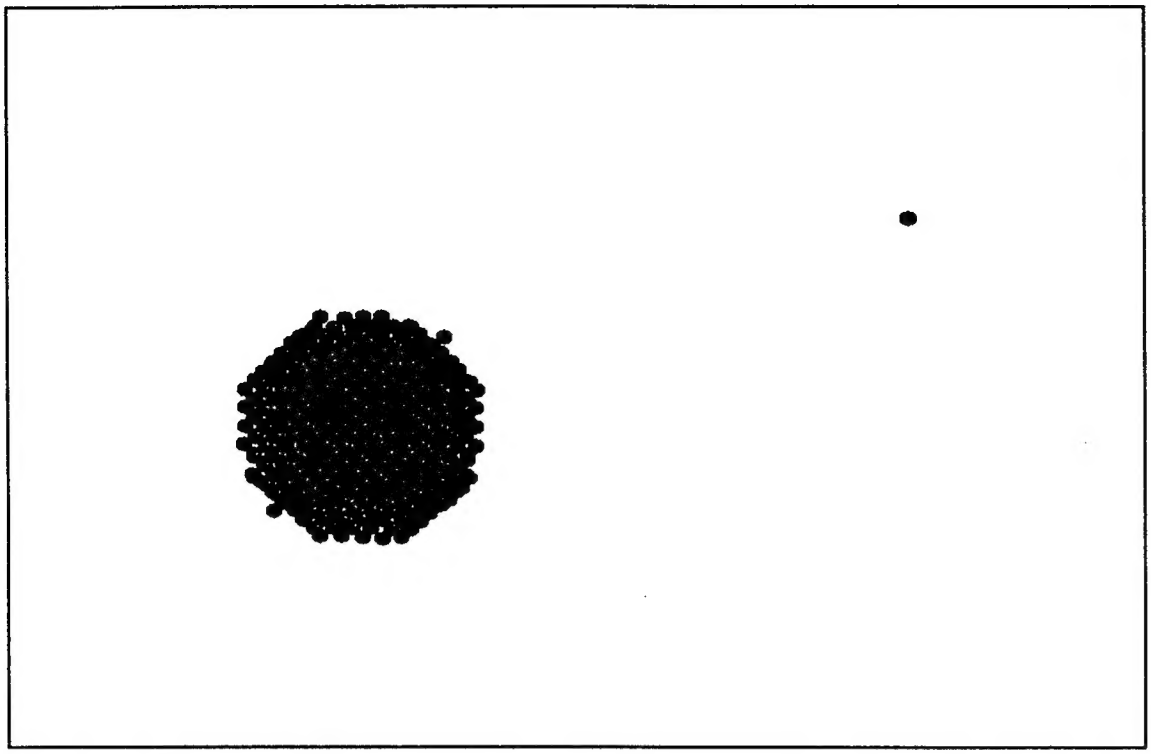


Fig.3 Evaporation of Cu system due to overheating.

To solve this we tagged only the outer shell of each system (482 atoms) and took RMSD data on only that shell to determine surface melting. We graphed the RMSD data of these runs with both the copper and the copper/gold systems to produce a spread of data around the surface melting point. These graphs helped use to find the approximate surface melting temperatures of each system (see Figures 4 and 5).

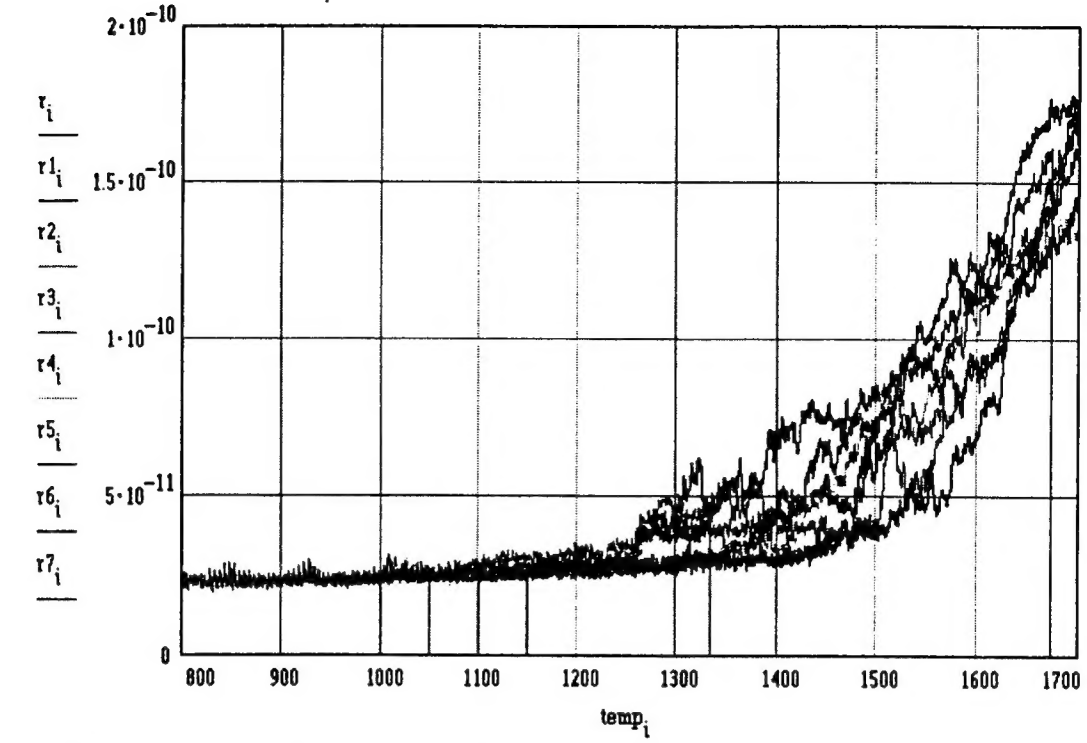


Fig.4 RMSD vs. temperature for outer shell (482-atoms) of 1289-atom Au-Cu system.

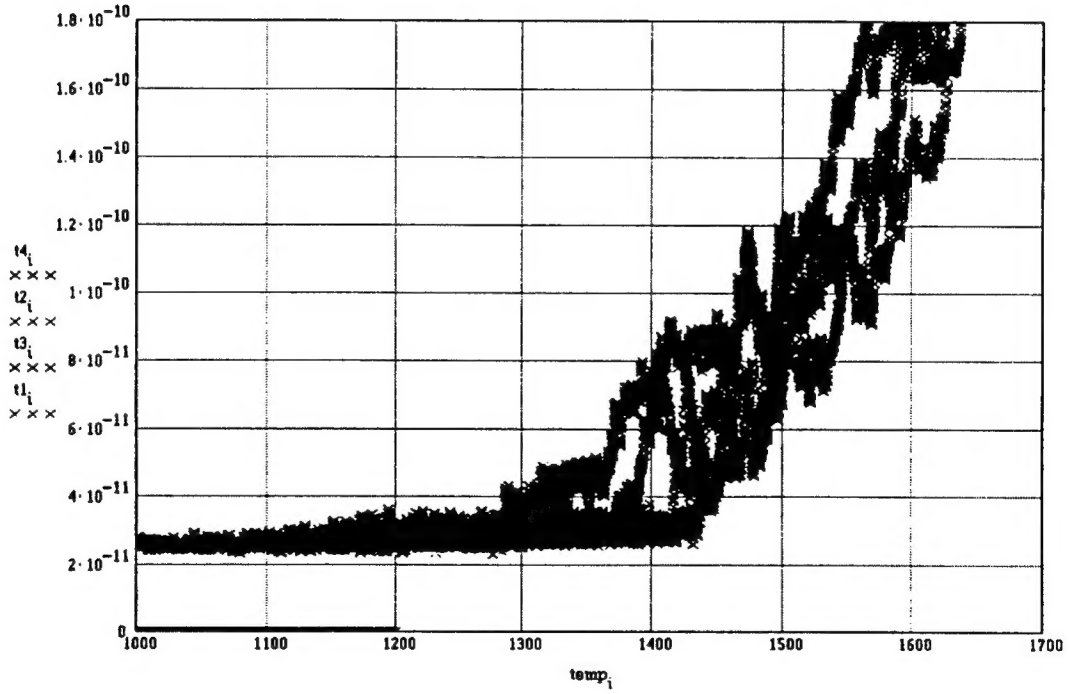


Fig.5 RMSD vs. temperature for outer shell (482-atoms) of 1289-atom Cu system.

Conclusions

After producing many on the Copper and Copper/Gold Wulff polyhedrons, it appears that the RMS deviation versus temperature of the Copper/Gold polyhedron "jumps" at a slightly lower temperature (on the average) then that of a Copper polyhedron. The difference between the two melting points is not as great or as clearly distinguish \able as we hoped for two reasons. First, (as stated before) the temperatures of such small systems as the 1289-atom Wulff polyhedron are only defined within ±60 K. A second consideration is that the melting of bulk that the eutectic curve is based on may behave differently than the surface melting we calculated.

Base Page Number 25

---

[1] Mdem was written at Rome laboratory. For more information contact Dr. Herb Helbig at Rome Laboratory, 525 Brooks Road, Rome, NY 13441-4505 or helbigh@rl.af.mil.
[2] Rasmol is freeware available via ftp.dcs.ed.ac.uk or sunsite.doc.ic.ac.uk.
[3] S. Valkealahti and M. Manninen, Computational Materials Science, 1 (1993) 123 - 134.